ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA DE TELECOMUNICACIÓN
UNIVERSIDAD POLITÉCNICA DE CARTAGENA

Proyecto Fin de Carrera

Asistente para discapacitados sobre arquitectura XScale en dispositivos personales con tecnología Wifi.

AUTOR: Félix Belzunce Arcos
DIRECTOR: Javier Vales Alonso
Septiembre / 2005

Autor	Félix Belzunce Arcos
E-mail del Autor	fbelzunc@gmail.com
Director(es)	Javier Vales Alonso
E-mail del Director	jvales@upct.es
Codirector(es)	-
Título del PFC	Asistente para discapacitados sobre arquitectura XScale en dispositivos personales con tecnología Wifi.
Descriptores	XScale, XML, Wifi, discapacitados

Resumen

En la actualidad el discapacitado se encuentra con numerosos obstáculos a la hora de desenvolverse. Tiene que ir tanteando por la ciudad para poder realizar tareas habituales en el día a días de las personas. Un ejemplo es a la hora de cruzar una calle regulada por un semáforo. Las ciudades no están provistas de infraestructuras adaptadas a estas personas.

El objetivo del proyecto es la creación de un sistema a través de hardware convencional para regular un semáforo con zumbador en una vía. En el sistema existen perfiles (fichas XML), que se encuentra en dispositivos móviles inalámbricos tipo móvil o PDA. Estas fichas son capturadas por el gestor del semáforo de forma automática a través de Bluetooh o Wifi. Una vez que han llegado, el gestor toma la decisión adecuada para adaptar la temporización del semáforo y el uso o no del zumbador en función de las deficiencias que existan en ese momento dentro del sistema.

La versión implementada en este proyecto consta por un lado de la aplicación cliente para PDAs+Wifi con sistema operativo Linux. Y, por otro lado, del módulo gestor encargado de tomar las decisiones adecuadas para la regulación del semáforo con zumbador ante la llegada de los discapacitados. Asimismo, se desarrollo una placa prototipo del semáforo que es controlada a través de la interfaz del puerto paralelo de una plataforma basada en XScale.

Titulación	Ingeniería Técnica de Telecomunicación, especialidad Telemática
Intensificación	-
Departamento	Tecnologías de la Información y las Comunicaciones
Fecha de Presentación	Septiembre – 2005

A mi familia, porque es uno de los pocos apoyos incondicionales con los que se puede contar. A mis amigos y compañeros de proyecto Juan Manuel Pérez Mañogil y Marco A. Esparza García con los que he compartido los momentos malos y buenos de este proyecto. A las maestras de taller de la UPCT (Noelia y Marga) , a David Henarejos y a nuestro director de proyecto Javier Vales por su gran ayuda. A todos los demás amigos, compañeros, y a todo aquel que de manera directa o indirecta a hecho posible la realización de este proyecto, y de lo que es más importante, de mi formación como persona y estudiante. A todos ellos, gracias.

Félix Belzunce Arcos

Índice General

Capítulo 1

Capitulo 2

Capítulo 3

Capítulo 4

Capítulo 5

Índice de gráficos

Capítulo 1

Introducción

1.1 Planteamiento

En la sociedad actual, los discapacitados sufren una gran discriminación frente al resto de ciudadanos. A pesar de los esfuerzos desplegados en los últimos años, la mayoría de los accesos y aplicaciones urbanas siguen básicamente pensadas para el ciudadano medio, mientras que los discapacitados físicos se ven discriminados, y han de desplegar grandes esfuerzos para intentar desarrollar una vida normal en ciudades que no disponen de medios o éstos no son suficientes para atender sus necesidades.

Esto ocurre en particular con las personas que sufren discapacidad visual. En una cuestión tan elemental y vital como es el paso de peatones que cuentan con semáforo, se ven en la necesidad de tantear hasta poder encontrar el botón que active el color verde que les franquea el paso. Por otro lado, los discapacitados motrices, una vez que pulsan dicho botón, apenas disponen de tiempo suficiente para cruzar la vía.

Una buena solución a este problema puede ser la de conectar las aplicaciones ya existentes en los semáforos a los discapacitados, de manera que las puedan manejar sin esfuerzo alguno. Idealmente esta solución debería requerir la menor cantidad de interacción por parte del usuario, es decir, que el proceso de aclimatar el entorno urbano al usuario se llevase acabo sin la intervención de éste. Por lo tanto, debe ser el medio el que analice a sus usuarios para tomar las decisiones apropiadas en cada caso, es decir, que va a ser el entorno el que se adapte a una colectividad de usuarios.

Evidentemente, para que el entorno pueda analizar a los usuarios en un momento dado, éstos deben llevar alguna clase de identificador con información personal vital para la toma de decisión por parte de aquél. Como tenemos como requisito que la interacción entre el usuario y el entorno sea lo más automática posible, hemos de descartar remedios típicos como tarjetas de identificación.

Actualmente, las tecnologías inalámbricas están en auge y no es difícil llegar a la conclusión de que esta respuesta puede ser la acertada. La mayoría de los usuarios disponen de algún tipo de dispositivo de comunicación inalámbrica, ya sea móvil, PDA u ordenador portátil. Aprovechando esta característica para solucionar el problema planteado, podríamos desarrollar una aplicación para dispositivos móviles que fuese el que interactuara con el medio mediante tecnologías inalámbrica actuales, principalmente Bluetooth para dispositivos móviles, Wi-Fi para PDA's y ordenadores portátiles.

1.2 Tecnologías inalámbricas

Bluetooth y Wi-fi son tecnologías inalámbricas que actualmente llevan incorporado la mayoría de dispositivos móviles que están en el mercado. El bajo coste, los distintos niveles de seguridad que llevan incorporados, el gran ancho de banda, el bajo consumo así como las altas prestaciones que nos proporcionan estos dispositivos hacen que sean ideales para nuestro propósito.

El sistema que se propone ha sido desarrollado según estas dos tecnologías, no sólo por todo lo anterior, sino también porque su modo de funcionamiento ofrece la posibilidad de que tan sólo, a través del encendido del dispositivo móvil, el sistema empiece a funcionar sin necesidad alguna de actuación por parte del discapacitado, de aquí su gran ventaja.

Tanto Bluetooth como Wi-fi funcionan en las llamadas LAN´s por espectro ensanchado, construidos habitualmente de manera celular en las que celdas adyacentes utilizan distintas frecuencias para evitar interferencias. Los dispositivos suelen utilizar antenas omnidireccionales con una potencia máxima de transmisión de 1 Watio, aunque dicha configuración no es obligatoria. Utilizan la banda sin licencia de 2.4 Ghz, es la llamada banda ISM (Industrial, Scientific and Medical).

Debido a que tanto Bluetooth como Wi-fi funcionan en la misma banda de frecuencia las pérdidas de transmisión se verán acentuadas, por ello se hacía necesario un protocolo orientado a conexión con el que conseguir minimizar la pérdidas de datos en transmisión y que nos aportara fiabilidad en la recepción de los datos. Por todo lo anterior y porque además son protocolos estándares totalmente probados y fiables hemos hecho uso de TCP/IP para Wi-fi y RFCOMM para

Bluetooth.

1.3 Aplicaciones de las tecnologías en personas discapacitadas

Nuestro objetivo es la realización de una aplicación para dar soporte a personas con alguna discapacidad de tipo locomotriz o visual, de tal forma que podamos facilitar algo tan común en el día a día como el mero hecho de cruzar una vía regulada por un semáforo.

Para la realización del proyecto contamos con un proceso servidor capaz de regular de forma inteligente el funcionamiento de un semáforo. El proceso servidor estará a la espera de la llegada de clientes dentro de la zona de cobertura y tras la conexión, estos dispositivos enviarán al servidor una ficha con unos determinados datos del cliente acerca de la discapacidad que tiene y en función de estas características el servidor tomará una decisión u otra.

Cabe la posibilidad de que varios discapacitados accedan simultáneamente al sistema, en este caso el servidor es capaz de tomar una decisión coherente al número y tipo de discapacitados que se encuentren en ese momento.

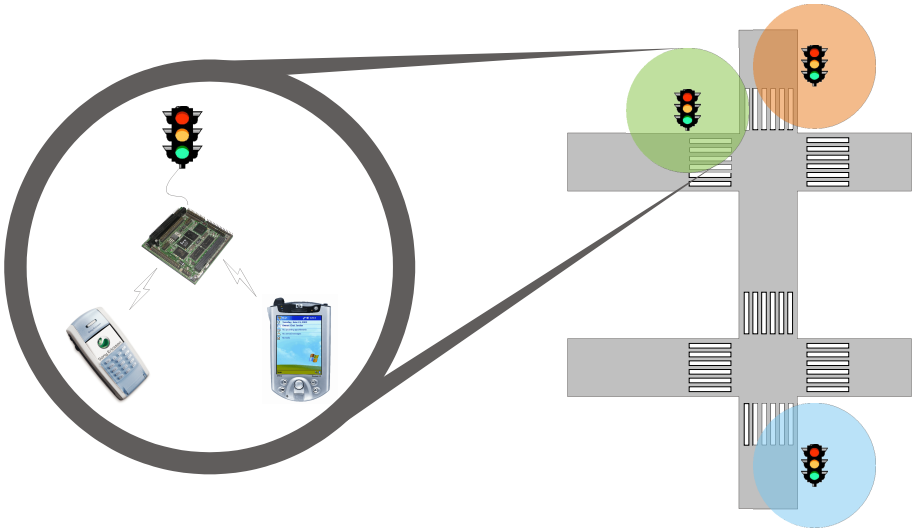


Figura 1.1 Esquema lógico de la disposición del sistema

1.4 Arquitectura general del sistema

Se describirá el funcionamiento general del sistema a partir de la figura . En ella se pueden distinguir los siguientes procesos:

•Cliente: Es aquel que sin ningún tipo de ínter actuación por parte del usuario del dispositivo móvil en el que corre dicho proceso, es capaz de forma automática de

establecer una conexión con el proceso servidor para enviarle una ficha XML (en el que se indica el tipo de discapacidad del usuario) y posteriormente detectar cuándo ha salido de la zona de cobertura para volver a establecer conexión con un nuevo proceso servidor.

•Servidor: El proceso servidor es el encargado de recibir las fichas del proceso cliente, enviársela al procesador XML y de forma parecida a como el cliente debe de detectar que ha salido de la zona de cobertura, el proceso servidor también debe de detectar cuando un cliente a abandonado la zona de cobertura que cubre.

•Procesador XML: Proceso que de forma local establece una comunicación con el proceso toma de decisión indicándole a éste el tipo o tipos de discapacidad que ha extraído de la ficha que le ha pasado el proceso servidor.

•Proceso toma de decisión: El proceso toma de decisión tras recibir el tipo de discapacidad del individuo o individuos que en un momento dado se encontraban dentro de la zona de cobertura, es capaz de tomar una buena decisión a la hora de controlar la placa que simula un semáforo con un zumbador.

Al final todos estos procesos deben de comunicarse con un determinado orden de manera que la placa funcione conforme se pretende. Las principales interacciones entre los procesos son:

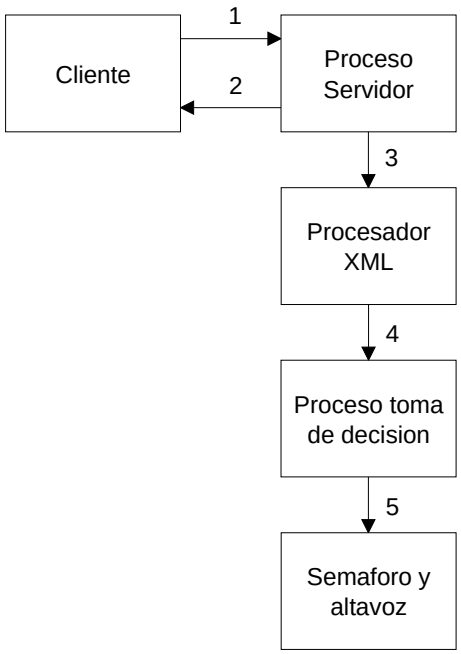


Figura 1.2.- Esquema básico del sistema

1. El cliente establece una conexión con el servidor y le envía su ficha XML.

2. El servidor indica al cliente si la ficha es válida.

3. Si la ficha es válida el servidor le pasa la ficha al procesador XML.

4. Proceso XML envía el tipo de discapacidad extraído de la ficha al proceso toma de decisión.

5. Proceso toma de decisión interactúa con la placa en función de los distintos tipos de discapacidades que podía haber en un determinado momento dentro de la zona de cobertura del servidor.

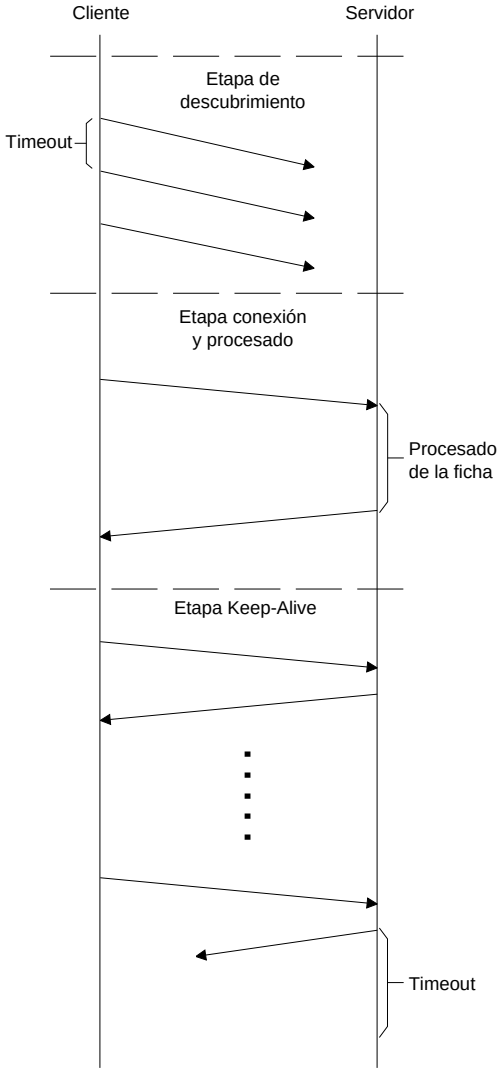


Figura 1.3. Interacción Cliente-Servidor

En el gráfico de la izquierda se muestra en detalle la interacción entre el cliente y el servidor:

El cliente se mantiene en un estado de descubrimiento en el que intenta hacer conexiones a un posible servidor cada cierto tiempo.

En el momento que se produce una conexión exitosa, se pasa a la siguiente etapa e inmediatamente envía su ficha XML, la cual es procesada por el servidor, recibiendo el cliente una respuesta sobre la validez de la ficha.

Si la ficha es correcta se procederá a entrar en una etapa para mantener viva la conexión. Cuando la comunicación con el servidor se corte, se entenderá que el cliente ha salido de cobertura, teniendo que volver al principio del diagrama.

1.5 Objetivos del proyecto

El objetivo de este proyecto ha sido dotar al discapacitado de un sistema capaz de funcionar sin interacción alguna por parte del usurario, en el que el entorno se adapta a las necesidades de cada uno de los discapacitados que puedan acceder al sistema.

Capitulo 2

Trabajos Relacionados

En este capítulo presentamos algunos sistemas que tienen como objetivo el de hacer un poco más fácil la vida de las personas discapacitadas. Presentamos los puntos más importantes de cada uno de ellos, destacando aspectos comunes y posibles mejoras.

■ Automation and Telematics for assisting people living at home

En este proyecto desarrollado por la Universidad de Helsinki se pretende usar la Telemática y la Automatización en beneficio de la personas mayores o con alguna minusvalía. Estos beneficios se llevan a cabo con la implantación en el domicilio de un conjunto de sistemas, intentado así dotar de completa autonomía a las personas discapacitadas en su hogar. Estos sistemas se conectan a un ordenador central encargado de recopilar información de todos los dispositivos y actuar según lo previsto. Algunos de estos dispositivos son cámaras, sensores térmicos, detectores magnéticos y de movimiento, robot móvil, etc... El sistema también ha sido desarrollado para ser controlado a través de un móvil o PDA e informar al usuario, a través de estos dispositivos, de los sucesos acontecidos dentro del hogar.

■ Accesibilidad para Discapacitados a través de Teléfonos y Servicios Móviles Adaptables

En este trabajo desarrollado en la Universidad de Deusto (Bilbao) se estudian los mecanismos que añaden accesibilidad a las personas discapacitadas en el uso de

dispositivos móviles. Estos mecanismos son el uso de sintetizadores de voz para personas invidentes, incorporación de botones especiales con números de marcación automática (por ejemplo el de emergencias 112), aplicaciones para el uso de los dispositivos inalámbricos como controles remotos, reconocedores de cadenas de texto o códigos especiales por medios de capturas fotográficas, para su posterior representación en audio por medio de los ya comentados sintetizadores de voz.

Dado que todos los sistemas desarrollados para mejorar la vida de las personas con alguna discapacidad cumplen un gran papel social es muy difícil encontrarles pegas, si bien el kit de la cuestión reside más en la posible interacción de los distintos sistemas para un funcionamiento compatible entre si. Es decir, sería ideal que tanto este sistema desarrollado por la Universidad de Helsinki que dota al usuario de total infraestructura dentro de su hogar, como el desarrollado por la Universidad de Deusto que dota a los dispositivos móviles de una mayor accesibilidad a personas discapacitadas, pudiesen interaccionar con el nuestro. Un posible ejemplo de esta interacción lo podemos encontrar cuando la persona discapacitada llega a casa y tiene que buscar la llave que abre la puerta, pues bien, este problema se podría solucionar con la colocación de un receptor en la puerta, capaz de comunicarse con el dispositivo inalámbrico, de la misma forma que en la elaboración de este proyecto se ha hecho para la comunicación con el semáforo. De esta forma la puerta se abriría automáticamente pudiendoselo comunicar al usuario por medio del sintetizador de voz.

Capítulo 3

Wireless Lan

3.1 Introducción a Wireless Lan

En los últimos años se ha producido un crecimiento espectacular en lo referente al desarrollo y aceptación de las comunicaciones móviles y en concreto de las redes de área local (Wireless LANs). La función principal de este tipo de redes es la proporcionar conectividad y acceso a las tradicionales redes cableadas (Ethernet, Token Ring...), como si de una extensión de éstas últimas se tratara, pero con la flexibilidad y movilidad que ofrecen las comunicaciones inalámbricas. El momento decisivo para la consolidación de estos sistemas fue la conclusión del estándar IEEE 802.11.

Las redes WLAN no pretenden sustituir a las tradicionales redes cableadas, sino más bien complementarlas. En este sentido el objetivo fundamental de las redes WLAN es el de proporcionar las facilidades no disponibles en los sistemas cableados y formar una red total donde coexistan los dos tipos de sistemas.

3.2 Definición de red de área local inalámbrica

Una red de área local inalámbrica puede definirse como a una red de alcance local que tiene como medio de transmisión el aire. Por red de área local entendemos una red que cubre un entorno geográfico limitado, con una velocidad de transferencia de datos relativamente alta (mayor o igual a 1 Mbps tal y como especifica el IEEE), con baja tasa de errores y administrada de forma privada. Por red inalámbrica

entendemos una red que utiliza ondas electromagnéticas como medio de transmisión de la información que viaja a través del canal inalámbrico enlazando los diferentes equipos o terminales móviles asociados a la red.

En las redes tradicionales cableadas esta información viaja fundamentalmente a través de cables coaxiales, pares trenzados o fibra óptica. Una red de área local inalámbrica, también llamada wireless LAN (WLAN), es un sistema flexible de comunicaciones que puede implementarse como una extensión o directamente como una alternativa a una red cableada. Este tipo de redes utiliza tecnología de radiofrecuencia minimizando así la necesidad de conexiones cableadas. Este hecho proporciona al usuario una gran movilidad sin perder conectividad.

El atractivo fundamental de este tipo de redes es la facilidad de instalación y el ahorro que supone la supresión del medio de transmisión cableado. Aún así, debido a que sus prestaciones son menores en lo referente a la velocidad de transmisión que se sitúa entre los 2 y los 54 Mbps frente a los 10 y hasta los 100 Mbps ofrecidos por una red convencional, las redes inalámbricas son la alternativa ideal para hacer llegar una red tradicional a lugares donde el cableado no lo permite, y en general las WLAN se utilizarán como un complemento de las redes fijas.

3.3 Aplicaciones de los sistemas WLAN

Las aplicaciones más típicas de las redes de área local que podemos encontrar actualmente
son las siguientes:

■ Implementación de redes de área local en edificios históricos, de difícil acceso y en general en entornos donde la solución cableada es inviable.

■ Posibilidad de reconfiguración de la topología de la red sin añadir costes adicionales. Esta solución es muy típica en entornos cambiantes que necesitan una estructura de red flexible que se adapte a estos cambios.

■ Redes locales para situaciones de emergencia o congestión de la red cableada. Estas redes permiten el acceso a la información mientras el usuario se encuentra en movimiento. Habitualmente esta solución es requerida en hospitales, fábricas, almacenes...

■ Generación de grupos de trabajo eventuales y reuniones ad-hoc. En estos casos no merece la pena instalar una red cableada. Con la solución inalámbrica es viable implementar una red de área local aunque sea para un plazo corto de tiempo.

■ En ambientes industriales con severas condiciones ambientales este tipo de redes sirve para interconectar diferentes dispositivos y máquinas.

■ Interconexión de redes de área local que se encuentran en lugares físicos distintos. Por ejemplo, se puede utilizar una red de área local inalámbrica para interconectar dos o más redes de área local cableadas situadas en dos edificios distintos.

3.4 IEEE 802.11

En este estándar se encuentran las especificaciones de las dos capas mas bajas de la capa OSI que hay que tener en cuenta a la hora de implementar una red de área local inalámbrica.

La norma 802.11 ha sufrido diferentes extensiones sobre la norma para obtener modificaciones y mejoras. De esta manera, tenemos las siguientes especificaciones:

■ 802.11 Especificación para 1-2 Mbps en la banda de los 2.4 GHz, usando salto de frecuencias(FHSS) o secuencia directa (DSSS).
■ 802.11b Extensión de 802.11 para proporcionar 11Mbps usando DSSS.
■ Wi-Fi (Wireless Fidelity) Promulgado por el WECA para certificar productos 802.11b capaces de inter operar con los de otros fabricantes.
■ 802.11a Extensión de 802.11 para proporcionar 54Mbps usando OFDM.
■ 802.11g Extensión de 802.11 para proporcionar 20-54Mbps usando DSSS y OFDM. Es compatible hacia atrás con 802.11b. Tiene mayor alcance y menor consumo de potencia que 802.11a.

3.4.1 Arquitectura lógica

El grado de complejidad de una red de área local inalámbrica es variable, dependiendo de las necesidades a cubrir y en función de los requerimientos del sistema que queramos implementar podemos utilizar dos tipos de configuración: **infrastructure** o **peer-to-peer** (Ad-Hoc) mode.

3.4.1.1 Modo de configuración Infraestructura

El portátil o dispositivo inteligente, denominado "estación" en el ámbito de las redes LAN inalámbricas, primero debe identificar los puntos de acceso y las redes disponibles. Este proceso se lleva a cabo mediante el control de las tramas de señalización procedentes de los puntos de acceso que se anuncian a sí mismos o mediante el sondeo activo de una red específica con tramas de sondeo.

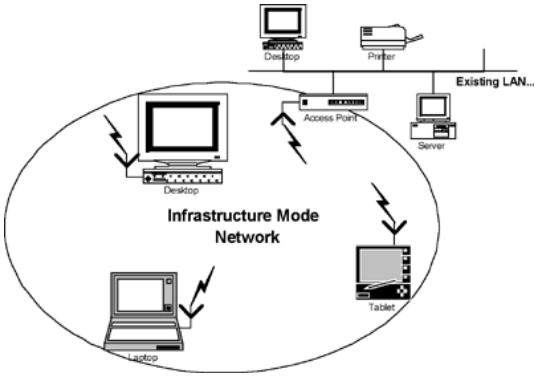


Figura 3.1. Topología del modo de funcionamiento Infraestructura

La estación elige una red entre las que están disponibles e inicia un proceso de autenticación con el punto de acceso. Una vez que el punto de acceso y la estación se han verificado mutuamente, comienza el proceso de asociación.

La asociación permite que el punto de acceso y la estación intercambien información y datos de capacidad. El punto de acceso puede utilizar esta información y compartirla con otros puntos de acceso de la red para diseminar la información de la ubicación actual de la estación en la red. La estación sólo puede transmitir o recibir tramas en la red después de que haya finalizado la asociación.

En la modalidad de infraestructura, todo el tráfico de red procedente de las estaciones inalámbricas pasa por un punto de acceso para poder llegar a su destino en la red LAN con cable o inalámbrica.

El acceso a la red se administra mediante un protocolo que detecta las portadoras y evita las colisiones. Las estaciones se mantienen a la escucha de las transmisiones de datos durante un período de tiempo especificado antes de intentar transmitir (ésta es la parte del protocolo que detecta las portadoras). Antes de transmitir, la estación debe esperar durante un período de tiempo específico después de que la red está despejada. Esta demora, junto con la transmisión por parte de la estación receptora de una confirmación de recepción correcta, representan la parte del protocolo que evita las colisiones. Observe que, en la modalidad de infraestructura, el emisor o el receptor es siempre el punto de acceso.

Dado que es posible que algunas estaciones no se escuchen mutuamente, aunque ambas estén dentro del alcance del punto de acceso, se toman medidas especiales para evitar las colisiones. Entre ellas, se incluye una clase de intercambio de reserva que puede tener lugar antes de transmitir un paquete mediante un intercambio de tramas "petición para emitir" y "listo para emitir", y un vector de asignación de red que se mantiene en cada estación de la red. Incluso aunque una estación no pueda oír la transmisión de la otra estación, oirá la transmisión de "listo para emitir" desde el punto de acceso y puede evitar transmitir durante ese intervalo.

El proceso de movilidad de un punto de acceso a otro no está completamente definido en el estándar. Sin embargo, la señalización y el sondeo que se utilizan para buscar puntos de acceso y un proceso de reasociación que permite a la estación asociarse a un punto de acceso diferente, junto con protocolos específicos de otros fabricantes entre puntos de acceso, proporcionan una transición fluida.

La sincronización entre las estaciones de la red se controla mediante las tramas de señalización periódicas enviadas por el punto de acceso. Estas tramas contienen el valor de reloj del punto de acceso en el momento de la transmisión, por lo que sirve para comprobar la evolución en la estación receptora. La sincronización es necesaria por varias razones relacionadas con los protocolos y esquemas de modulación de las conexiones inalámbricas.

3.4.1.2 Modo de configuración Ad-hoc

En una topología ad hoc, los propios dispositivos inalámbricos crean la red LAN y no existe ningún controlador central ni puntos de acceso. Cada dispositivo se comunica directamente con los demás dispositivos de la red, en lugar de pasar por un controlador central. Esta topología es práctica en lugares en los que pueden reunirse pequeños grupos de equipos que no necesitan acceso a otra red. Ejemplos de entornos en los que podrían utilizarse redes inalámbricas ad hoc serían un domicilio sin red con cable o una sala de conferencias donde los equipos se reúnen con regularidad para intercambiar ideas.

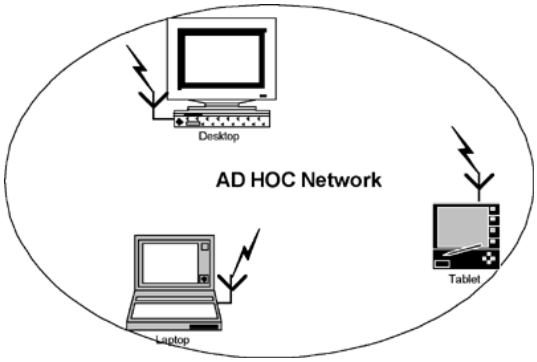


Figura 3.2. Topología del modo de funcionamiento Ad-Hoc

Las redes 802.11 pueden extenderse arbitrariamente, para ello se enlazan varios BSS, con lo que se forma lo que se llama área de servicio extendida (Extended Service Set, ESS). Las BSS se encadenan mediante una red de transporte. Esta red de transporte se denomina sistema de distribución (Distribution System, DS). La especificación no señala una tecnología en particular para realizar esta función, sólo requiere que proporcione una serie de servicios.

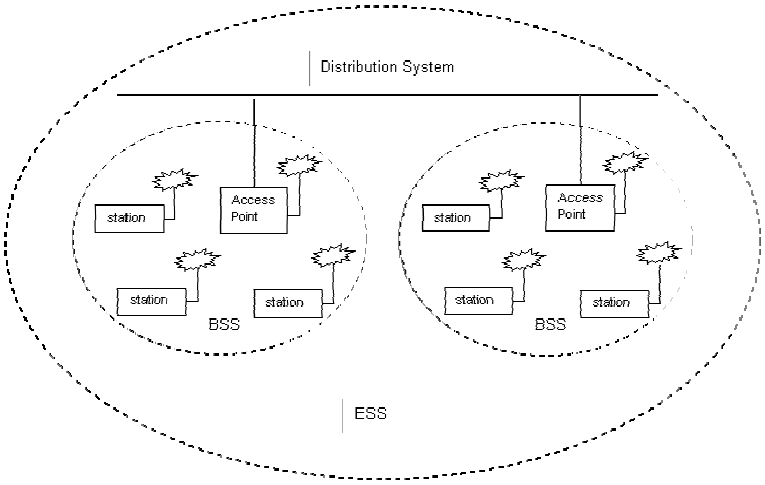


Figura 3.3. Extended Service Set (ESS)

La ESS se forma por la unión de varias BSS mediante el sistema de distribución. Las estaciones dentro de una misma ESS pueden comunicarse entre todas ellas, independientemente de si pertenecen a la misma BSS o a diferentes, o incluso si se están moviendo entre ellas. Una estación puede moverse de una BSS a otra, con lo que se asocia a un AP distinto, el sistema de distribución debe gestionar estas asociaciones, debe permitir que los AP comuniquen al resto de los AP qué estaciones están asociadas a ellos. Es necesario un método para realizar estas notificaciones. No hay, de momento, un método estandarizado, aunque el grupo 802.11 trabaja en ello.

3.4.2 Límites de movilidad

IEEE 802.11 soporta la movilidad de las estaciones garantizando el mantenimiento de la conexión en las transiciones entre BSS. Las estaciones monitorizan continuamente la calidad de la señal de todos los AP que han sido asignados a cubrir una ESS. Cuando una estación detecta una señal mejor de un AP distinto al que está asociado, termina su asociación con el primero y se asocia con el segundo. El DS permitirá que la estación reciba las tramas a través del nuevo AP. Se mantiene la conexión de nivel de enlace en todo momento. Sin embargo, no garantiza el mantenimiento de las conexiones entre ESS.

3.4.3 Servicios de red

Una forma de definir una tecnología es declarar los servicios que debe proporcionar

y permitir que los vendedores lo implementen de la manera que consideren adecuada. IEEE 802.11 define nueve servicios que deben proporcionar las redes inalámbricas, de tal manera que la funcionalidad que proporcionen sea equivalente a la de una LAN de cable.

Los servicios se describen a continuación:

■ Distribución. Lo usan las estaciones para transmitir tramas en redes de infraestructura. Una vez que la trama ha sido aceptada por el AP, usa el sistema de distribución para entregar la trama a la estación destino.

■ Integración. Se encarga de la función de pasarela con otros sistemas IEEE802.x. En concreto, define el componente portal que se encargará de aspectos necesarios como redireccionamiento.

■ Asociación. Servicio necesario para que una estación pueda adherirse al modo infraestructura y utilizar sus servicios.

■ Reasociación. Permite que una asociación se transfiera de un AP a otro. La reasociación la inicia una estación cuando las condiciones de la señal indican que asociarse con otro AP sería beneficioso. Cuando la reasociación termina, el DS actualiza sus registros de localización.

■ Disociación. Para terminar una asociación. Es una notificación y por tanto no puede ser rechazada. Las estaciones deberían disociarse al abandonar la red.

■ Autenticación y Deautentificación. La autenticación se usa para establecer la identidad una estación. El estándar soporta diferentes métodos de autenticación, desde la autenticación de sistema abierto (método por defecto, pero inseguro) hasta sistemas de clave pública, aunque no obliga a usar ninguno en particular. Este servicio se usa obligatoriamente antes de establecer la asociación. El estándar especifica el servicio complementario, es decir, la finalización de una relación autenticada.

■ Privacidad. Este servicio utilizará WEP para el encriptado de los datos en el medio.

■ Entrega de tramas (MSDU) entre STAs. Es el servicio básico que implementa toda estación: la transferencia de datos al destino.

3.4.4 Nivel de acceso al medio MAC

Los diferentes métodos de acceso de IEEE802 están diseñados según el modelo OSI y se encuentran ubicados en el nivel físico y en la parte inferior del nivel de enlace

o subnivel MAC.

Además, la capa de gestión MAC de 802.11 cubre tres áreas funcionales: **control de acceso**, **entrega fiable** y **seguridad**.

3.4.4.1 Descripción Funcional MAC

La arquitectura MAC del estándar 802.11 se compone de dos funcionalidades básicas que *controlan el acceso al medio*: la función de coordinación puntual (PCF) y la función de coordinación distribuida.

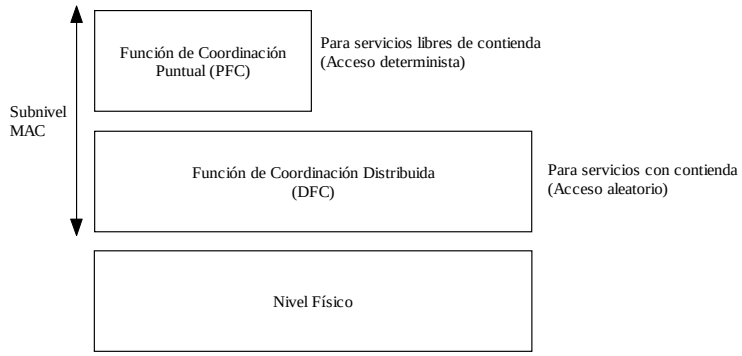


Figura 3.4. Descripción funcional Mac

3.4.4.1.1 DFC Función de Coordinación Distribuida

La función de coordinación es la funcionalidad que determina, dentro de un conjunto básico de servicios (BSS), cuándo una estación puede transmitir y/o recibir unidades de datos de protocolo a nivel MAC a través del medio inalámbrico. En el nivel inferior del subnivel MAC se encuentra la función de coordinación distribuida y su funcionamiento se basa en técnicas de acceso aleatorias de contienda por el medio.

Las características de DFC las podemos resumir en estos puntos:

- Utiliza MACA (CSMA/CA con RTS/CTS) como protocolo de acceso al medio
- Necesario reconocimientos ACKs, provocando retransmisiones si no se recibe
- Usa campo Duration/ID que contiene el tiempo de reserva para transmisión y ACK. Esto quiere decir que todos los nodos conocerán al escuchar cuando el canal volverá a quedar libre
- Implementa fragmentación de datos
- Concede prioridad a tramas mediante el espaciado entre tramas (IFS)
- Soporta Broadcast y Multicast sin ACKs

3.4.4.1.2 PFC Función de Coordinación Puntual

Por encima de la funcionalidad DCF se sitúa la función de coordinación puntual, PCF, asociada a las transmisiones libres de contienda que utilizan técnicas de acceso deterministas. El estándar IEEE 802.11, en concreto, define una técnica de interrogación circular desde el punto de acceso para este nivel. Esta funcionalidad está pensada para servicios de tipo síncrono que no toleran retardos aleatorios en el acceso al medio.

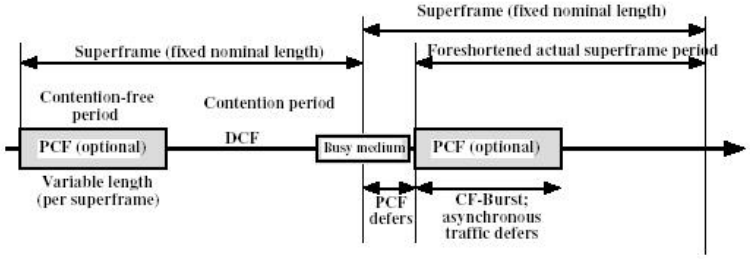


Figura 3.5. Intervalos de Supertrama

Estos dos métodos de acceso pueden operar conjuntamente dentro de una misma celda o conjunto básico de servicios dentro de una estructura llamada supertrama. Al comienzo de la supertrama el AP sondea las estaciones (aquellas configuradas para usar PCF) siguiendo un esquema round-robin. Les envía los datos dirigidos a ellas y pregunta si tienen algo que transmitir. Para asegurar la captura del medio, el AP establece la duración del intervalo sin contienda mediante el NAV[1]. El resto del tiempo se dedica a tráfico asíncrono. El intervalo de supertrama tiene una duración nominal fija. Al final de este intervalo, el AP lucha por el medio usando del PIFS[2]. Si el medio está ocupado, el AP espera antes de empezar el periodo sin contienda, que estará acortado en el siguiente intervalo de supertrama, por tanto.

El nodo utilizará una trama para la configuración de la supertrama, llamada Beacon, donde establecerá una CFRate o tasa de periodos de contienda. Pese a que el periodo de contienda se puede retrasar por estar el medio ocupado, la tasa se mantendrá en el siguiente periodo con medio libre.

3.4.4.2 Protocolo de Acceso al medio CSMA/CA

El algoritmo básico de acceso a este nivel es muy similar al implementado en el estándar IEEE 802.3 y es el llamado CSMA/CA (Carrier Sense Multiple Access /

[1] (Network Allocation Vector) que mantendrá una predicción de cuando el medio quedará liberado

[2] Es un tipo de espaciado entre tramas o IFS. PIFS (Point Coordination Function IFS). De duración media. Es usado por el PCF para enviar tráfico sin contienda previa.

Collision Advoidance). Este algoritmo funciona tal y como describimos a continuación:

El mecanismo de CSMA/CA se basa en un conjunto de retardos que permiten un cierto esquema de prioridades. Se define un espacio entre tramas (Interframe Spacing, IFS), que es un determinado tiempo de retardo. En realidad, hay tres tiempos distintos, pero el algoritmo se explica mejor ignorando este detalle.

Las reglas del algoritmo son las siguientes:

1-. Una estación que quiere transmitir comprueba el medio. Esta comprobación es virtual y física. Es decir, mientras el NAV no sea cero, el medio estará ocupado. Cuando el NAV es cero se comprueba físicamente el medio. Si el medio está libre, la estación espera un tiempo igual al IFS. Si al finalizar este intervalo el medio continúa libre, transmite.

2-. Si el medio está ocupado (porque estaba ocupado o porque se ocupó durante el IFS) la estación abandona la transmisión y continúa monitorizando hasta que la transmisión en curso acabe.

3-. Cuando la transmisión acaba, la estación espera otro IFS. Si el medio permanece libre ese último intervalo, la estación espera un tiempo aleatorio (usa el algoritmo de retroceso exponencial binario, como en Ethernet) y de nuevo comprueba el medio. Si sigue libre, la estación puede transmitir. Si durante el tiempo de espera (backoff) el medio se ocupa, el contador se para y continúa al volver a desocuparse el medio.

Para asegurar la estabilidad del algoritmo, se utiliza exactamente el mismo algoritmo de espera que Ethernet: el algoritmo de retroceso exponencial binario. El tiempo se ranura, con un tiempo de slot que depende de la velocidad de transmisión, se elige un entero aleatorio y en función de él se transmite en un slot u otro. Cada vez que se reintenta la transmisión, el intervalo de slots
aumenta.

En CSMA/CD el reintento se produce cada vez que se detecta una colisión. Con un medio inalámbrico no se puede detectar la colisión, por tanto, se utiliza el mecanismo de reconocimiento para detectar errores. Como se comentó previamente, la transmisión de una trama y su reconocimiento es una operación atómica. Si el transmisor no recibe un ACK, incrementa el contador de reintentos. Esto implica que el intervalo del algoritmo de retroceso se extenderá en el siguiente intento. Al llegar a cierto número de reintentos, se descarta la trama.

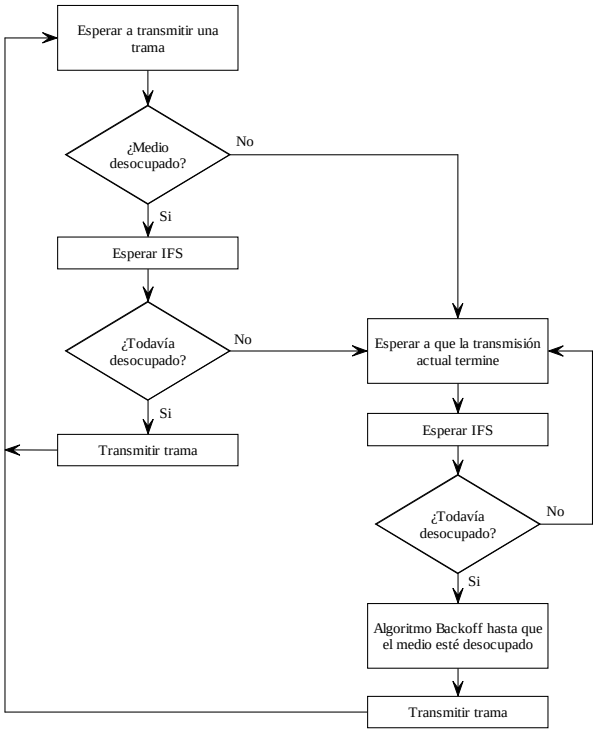


Figura 3.6. Algoritmo de acceso al medio

Sin embargo, CSMA/CA en un entorno inalámbrico y celular presenta una serie de problemas que intentaremos resolver con alguna modificación. Los dos principales problemas que podemos detectar son:

■ Nodos ocultos. Una estación cree que el canal está libre, pero en realidad está ocupado por otro nodo que no oye
■ Nodos expuestos. Una estación cree que el canal está ocupado, pero en realidad está libre pues el nodo al que oye no le interferiría para transmitir a otro destino.

La solución que propone 802.11 es MACA o MultiAccess Collision Avoidance. Según este protocolo, antes de transmitir el emisor envía una trama RTS[3] (Request to Send),

3.4.4.3 Espaciado entre tramas IFS

El tiempo de intervalo entre tramas se llama IFS. Durante este periodo mínimo, una estación STA estará escuchando el medio antes de transmitir. Se definen cuatro espaciados para dar prioridad de acceso al medio inalámbrico. Veámoslos de más cortos a más largos

[3] El mecanismo se explica en el apartado 3.4.4.5 Entrega Fiable de datos

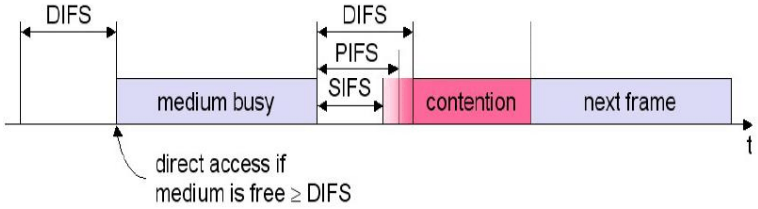


Figura 3.7. Espaciado entre tramas IFS

• **SIFS** (Short IFS). Este es el periodo más corto. Se utiliza fundamentalmente para transmitir los reconocimientos. También es utilizado para transmitir cada uno de los fragmentos de una trama. Por último, es usado por el PC o Point Control para enviar testigo a estaciones que quieran transmitir datos síncronos
• **PIFS** (Point Coordination Function IFS). Es utilizado por los PCF para ganar prioridad de acceso en los periodos libres de contienda. Lo utiliza el PCF para ganar la contienda normal, que se produce al esperar DIFS.
• **DIFS** (Distributed Coordination Function IFS). El IFS de mayor longitud. El tiempo mínimo que tiene que estar el canal libre para el acceso por contienda.
• **EIFS** (Extended IFS). Controla la espera en los casos en los que se detecta la llegada de una trama errónea. Espera un tiempo suficiente para que le vuelvan a enviar la trama u otra solución.

3.4.4.4 Conocimiento del medio

Las estaciones tienen un conocimiento específico de cuando la estación, que en estos momentos tiene el control del medio porque está transmitiendo o recibiendo, va a finalizar su periodo de reserva del canal.

Esto se hace a través de una variable llamada NAV (Network Allocation Vector) que mantendrá una predicción de cuando el medio quedará liberado.

Tanto al enviar un RTS como al recibir un CTS, se envía el campo Duration/ID con el valor reservado para la transmisión y el subsiguiente reconocimiento. Las estaciones que estén a la escucha modificarán su NAV según el valor de este campo Duration/ID. En realidad, hay una serie de normas para modificar el NAV, una de ellas es que el NAV siempre se situará al valor más alto de entre los que se disponga.

3.4.4.5 Entrega Fiable de datos

Para minimizar los problemas del medio inalámbrico se incorporan dos mecanismos. Por una parte, para evitar el problema de la pérdida de tramas en el canal, se

incorpora un mecanismo de reconocimiento positivo. Todas las tramas transmitidas deben ser reconocidas (mediante una trama ACK). Este intercambio se trata como una operación atómica y no puede ser interrumpido
por la transmisión de otra estación.

Por otra parte, para solucionar el problema de los nodos ocultos se utiliza un mecanismo de RTS/CTS. Las tramas RTS/CTS se transmiten antes de la trama de datos. Las estaciones que escuchan estas tramas ceden el medio durante un tiempo determinado. Además, la trama de datos debe ser reconocida, por tanto, la transmisión se produce en 4 pasos. Esto implica que se consume una cantidad considerable de capacidad, además de la latencia añadida. Este mecanismo se usa sólo en entornos de alta ocupación. El mecanismo se puede deshabilitar y además se puede controlar un umbral de RTS (RTS threshold). Sólo a las tramas cuyo tamaño sea mayor que el umbral se les aplica el mecanismo de RTS/CTS.

3.4.4.6 Fragmentación y reensamblado

A diferencia de Ethernet y de las tecnologías de nivel de enlace de datos, el MAC de 802.11 sí que fragmenta y reensambla paquetes. El canal inalámbrico tiene un nivel muy alto de interferencias y es preferible enviar paquetes cortos para evitar la retransmisión de tramas largas dañadas.

La fragmentación se produce cuando el tamaño de un paquete excede unumbral de fragmentación. Las tramas enviadas tienen el mismo número desecuencia y un número ascendente de fragmentos para permitir el reensamblado. Todos los paquetes que componen una trama se envían en una ráfaga de fragmentación, es decir, utilizando el SIFS después de cada ACK del receptor para cada fragmento. Además se utiliza el NAV para asegurar el control del medio.

El umbral de fragmentación suele coincidir con el umbral de RTS, es decir, que los paquetes fragmentados usan el mecanismo RTS/CTS. En definitiva, una estación captura el medio durante todo el tiempo necesario para enviar todos los fragmentos.

3.4.5 Formato de la trama

Para cubrir los requerimientos del medio inalámbrico, así como los servicios ofrecidos por 802.11 se tiene que aumentar la complejidad del formato de trama 802.11. El formato de trama se muestra en la siguiente figura El preámbulo lo incorpora la capa de nivel físico.

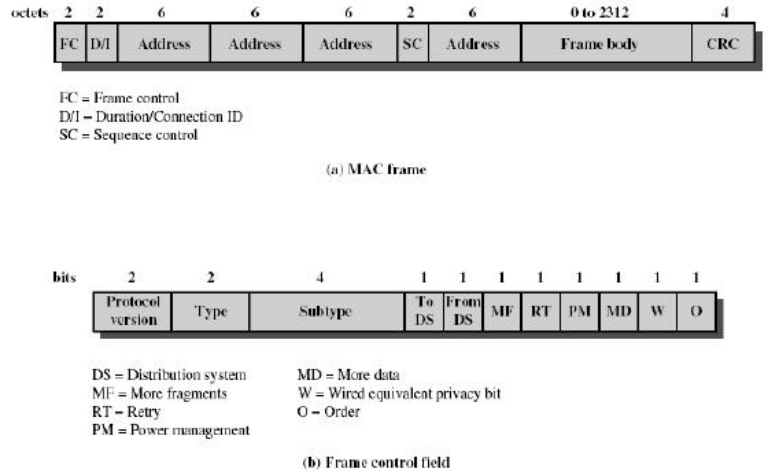


Figura 3.8. Formato de la trama 802.11

La característica más notable es que se puede usar hasta 4 campos de direcciones. No todas las tramas usan los 4 campos de direcciones y los valores asignados al campo varían en función del tipo de trama. Los campos que incluye son los siguientes:

■ Frame Control. Se utiliza para identificar el tipo de trama. Es necesario porque para ofrecer todos los servicios (asociación, disociación, etc.) hay que utilizar diferentes tipos de trama.
■ Type y subtype. Indican el tipo de trama usado.
■ ToDS y FromDS. Indican si una trama está destinada al sistema de distribución. La interpretación de los campos de direcciones depende de estos bits.
■ More fragments. Es necesario para indicar si quedan más fragmentos.
■ Retry. Si la trama es una retransmisión, para evitar duplicados en el receptor.
■ Power Management. Este bit indica si el origen pasará a modo de ahorro de potencia después de este intercambio.
■ More Data. Indica si quedan más datos para transmitir.
■ WEP. Indica si la trama ha sido procesada con el algoritmo WEP.
■ Order. Indica a la estación que las tramas deben ser procesadas en orden.
■ Duration/ID. Se utiliza, en función del tipo de trama, para establecer un valor de NAV o para indicar el identificador de asociación, es decir, la BSS a la que pertenece la estación.
■ Campos de direcciones. Las direcciones siguen el formato de 48 bits de IEEE 802. Los campos de dirección están numerados porque se usan diferentes campos para diferentes propósitos en función del tipo de trama.

La regla general sería que la Dirección 1 se usa para el receptor, la Dirección 2 para el transmisor y la Dirección 3 para filtrados del receptor. La mayoría de las

tramas de datos usan 3 campos, para destino, origen e ID de BSS respectivamente. El número y posición de los campos en una trama de datos depende de cómo la trama viaja en relación con el sistema de distribución.

■ Sequence Control. Este campo se usa para los números de secuencia, necesarios al usar reconocimiento, y para numerar los fragmentos, necesarios al utilizar fragmentación.
■ Cuerpo. Es el campo de datos. El tamaño máximo es de 2304 octetos.
■ Frame Check Sequence. Es el campo de control de errores. Consiste en un CRC de 32 bits.
Hay 3 tipos de tramas, que incluyen diferentes subtipos:
■ Tramas de control. Ayudan en la entrega fiable de tramas de datos.
■ Tramas de datos. Hay hasta 8 subtipos, organizados en dos grupos.
■ Tramas de gestión. Gestionan la comunicación entre AP y estaciones.

3.4.6 Unión a la red

Cuando una estación entra en funcionamiento, debe determinar si hay otra estación o un punto de acceso presente, para asociarse con ellos. La estación realiza un proceso de descubrimiento antes de empezar con la asociación. Hay dos formas de realizar el descubrimiento:

■ Rastreo pasivo (passive scanning). Para cada canal, la estación escucha durante un determinado tiempo. Cada AP emite señales de faro que incluyen el identificador de la BSS. Una vez detectada la BSS con el identificador deseado se inicia el proceso de autenticación y asociación.

■ Rastreo activo. La estación envía tramas sonda indicando el identificador de la red a la que quiere unirse. Entonces espera una trama de respuesta de la red a la que quiere unirse. Una estación también puede enviar una sonda con el identificar de red broadcast, que hará que todas las redes en la vecindad respondan. Responden los AP y, en el caso de redes ad-hoc, la última estación que generó una trama de faro.

3.4.7 Nivel físico

IEEE 802.11 define varios niveles físicos, en diversas especificaciones. Sus características resumidas son las siguientes:

■ 802.11. Define tres niveles físicos.
    ■ DSSS a 2.4 GHz con velocidades de 1 y 2 Mbps.
    ■ FHSS a 2.4 GHz con velocidades de 1 y 2 Mbps.
    ■ Infrarrojos entre 850 y 950 nm con velocidades de 1 y 2 Mbps.
■ 802.11b. Es el más usado. DSSS a 2.4 GHz con velocidad de 5.5 y 11 Mbps.

- 802.11g. El más reciente. DSSS a 2.4 GHz con velocidad de 54 Mbps.
- 802.11a. Modulación OFDM a 5 GHz con velocidad de 54 Mbps.

En la arquitectura del sistema la capa física se divide en dos subcapas: la Physical Layer Convergence Procedure (PLCP) y la Physical Medium Dependent (PMD) cuyas misiones son la detección de portadora, la transmisión y la recepción. La primera se encarga de añadir el preámbulo y la cabecera física. La segunda se encarga de transmitir y recibir los bits por la antena.

Para asegurar la compatibilidad entre versiones, la cabecera de nivel físico indica la velocidad a la que se transmitirán los datos. Entonces, todas las cabeceras (y preámbulo) se transmiten a 1 Mbps, independientemente de la velocidad a la que se transmitan luego los datos, es decir, la trama MAC.

La mayoría de los productos, si detectan un nivel alto de interferencias, reducen la velocidad de transmisión. Nivel físico 802.11 y 802.11b por espectro ensanchado Puesto que es el nivel físico más usado se describirán brevemente sus características. El nivel físico de 802.11b está basado en el nivel físico de espectro ensanchado de 802.11.

Para 802.11, la secuencia de ensanche es un código Barker de 11 chips {+1,-1,+1,+1,-1,+1,-1,-1,-1}. Esta secuencia se suma modulo-2 con la señal de datos. En función de la velocidad de los datos se utiliza una modulación distinta: para 1 Mbps se utiliza DBPSK y para 2 Mbps se utiliza DQPSK. Hay que prestar atención al hecho de que todos los equipos y redes usan la misma secuencia de Barker. Por tanto, no se está utilizando CDMA para el acceso al medio, sino simplemente técnicas de espectro ensanchado para reducir interferencias. Para conseguir altas velocidades, 802.11b ya no usa la secuencia de Barker, sino una codificación bastante compleja, denominada Complementary Code Keying (CCK), que posteriormente se modula con DQPSK.

Capítulo 4

XML

El XML (eXtended Markup Language) está llamado a ser el nuevo estándar en la gran telaraña de la World Wide Web, de momento este lenguaje ya ha recibido el apoyo de algunas de las compañías informáticas más importantes como IBM o Microsoft.

4.1 Introducción

El XML es un descendiente del SGML (Standard Generalised Markup Language), pero está orientado, al igual que el HTML (HyperText MarkUp Language) , a aplicaciones basadas en la Web. El XML está llamado a ser el sucesor del que hoy por hoy es el lenguaje más utilizado en la creación de documentos web: el HTML. Frente a éste tiene algunas ventajas, especialmente a nivel definición de etiquetas.

Pero lo que más hace destacar a este lenguaje frente a su antecesor es su carácter abierto, ya que al contrario que en HTML, donde las etiquetas vienen predefinidas, XML permite definir nuestras propias etiquetas. El lenguaje XML permite trabajar de una forma más óptima con los datos, ya que aporta un componente más a los que el HTML ya aportaba, las entidades (en inglés entities) que permite al sistema saber el tipo de dato con el que está trabajando. Esta nueva característica que aporta el XML, permitirá desarrollar un trabajo más efectivo a la hora de consultar una base de datos; esto a nivel intranet, ya que realmente para el entorno para el que ha sido pensado, Internet, permite, por ejemplo, mejorar de manera espectacular la búsquedas en dicho entorno.

Sobre todo en el campo del comercio electrónico, ya que permitirá que

nuestro sistema reciba la información sobre, por ejemplo, los tejidos disponibles en el mercado, aunque cada fabricante use un software de base de datos completamente distinto. Así podremos consultar las últimas novedades por ejemplo en música, recibiéndolas en nuestra base de datos personal sin tener que perder un tiempo precioso en pasar al formato debido la información que nos va llegando de cada tipo de sistema, y sin tener que instalar una aplicación para cada formato de fichero.

Son muchas las novedades que aporta este lenguaje frente a sus antecesores pero destacan las siguientes:

- Es un lenguaje diseñado de forma que sea más sencilla su programación a los desarrolladores.
- Al estar basado en el extendido lenguaje SGML, no se necesitan crear nuevas aplicaciones para utilizarlo, ya que éste posee un gran número de ellas.
- Todo en el XML ha sido pensado para el mejor aprovechamiento de la Red, es un lenguaje "maduro", que ha surgido fruto de la experiencia con HTML y SGML.
- Al contrario que HTML, en XML se pueden desarrollar nuevas etiquetas, definiéndolas previamente.
- En XML la interactividad entre programas, para que estos usen fragmentos de código unos de otros, es total.
- XML al igual que sus antecesores es un lenguaje "sin propietario", ya que se trata de un estándar internacional.
- Además permite usar distintos códigos de signos, logrando así abarcar todas las lenguas del planeta.

Antes de crear un fichero XML, debemos de crear un fichero DTD (Document Type Definition), en el cual se deben definir todas las etiquetas que se están utilizando. Después ese fichero XML si es correcto es validado, creándose así un fichero con extensión XML accesible, utilizando el estilo XSL, desde los navegadores, que soportan esta utilidad.

4.2 XML frente a SGML

El XML es ni más ni menos que una versión mejorada de SGML, como se ha dicho antes, al cual se han añadido nuevas características orientadas a hacer de éste un lenguaje óptimo para Internet. Al ser el XML un sucesor del SGML, éste puede utilizar todas las herramientas del anterior (que son bastantes), de modo que XML esté empezando a ser conocido de forma mucho más rápida que otros lenguajes.

	HTML/DHTML	XML	SGML
Gramática	Fija y no ampliable	Extensible	Extensible
Estructura	Monolítica	Jerárquica	Jerárquica
Nº de marcas	Fijas	Sin límite	Sin límite
Complejidad	Baja	Mediana	Alta
Diseño de páginas	Fijado por tags. Etiquetas con atributos CSS en DHTML	CSS o XSL	DSSSL
Enlaces	Simples enlaces	Poderosos enlaces (XLL)	HyTime
Exportabilidad (formatos/aplicaciones)	No	Sí	Sí
Validación	Sin validación	Pueden validarse	Obligatorio DTD
Búsquedas	Simple y a veces resuelta por *scripts* o CGI	Potente búsqueda. Con capacidad para personalizarla	Son posibles potentes búsquedas.
Indización/Catalogación de páginas web	Sólo lo permite los atributos de la etiqueta <META>, e implementaciones como DC.	Una descripción abierta y personalizable con el RDF.	Algún proyecto como TEI, DLI, etc.

4.3 Estructura de un documento XML

Lo primero que debemos saber es que hay dos tipos de documentos XML: válidos y bien formados. Éste es uno de los aspectos más importantes de este lenguaje, así que hace falta entender bien la diferencia:

- Bien formados
  Son todos los que cumplen las especificaciones del lenguaje respecto a las reglas sintácticas que después se van a explicar, sin estar sujetos a unos elementos fijados en un DTD (luego veremos lo que es un DTD). De hecho los documentos XML deben tener una estructura jerárquica muy estricta, de la que se hablará más tarde, y los documentos bien formados deben cumplirla.
- Válidos
  Además de estar bien formados, siguen una estructura y una semántica determinada por un DTD: sus elementos y sobre todo la estructura jerárquica que define el DTD, además de los atributos, deben ajustarse a lo que el DTD dicte.

En un DTD se lleva acabo la definición de los elementos que puede haber en el documento XML, y su relación entre ellos, sus atributos, posibles valores, etc. De hecho DTD son las siglas de Document Type Definition, o Definición de Tipo de Documento. En definitiva, es una especie de definición de la gramática del

documento.

Es importante entender que cuando se procesa cualquier información formateada mediante XML, lo primero es comprobar si está bien formada, y luego, si incluye o referencia a un DTD, comprobar que sigue sus reglas gramaticales. Hay pues diferencia entre los parsers que procesan documentos XML sin comprobar que siguen las reglas marcadas por un DTD (sólo comprueban que está bien formado), que se llaman parsers no validadores, y los que sí lo hacen, que son parsers validadores (comprueba que además de bien formado, se atiene a su DTD y es válido).

A continuación podemos ver un ejemplo de documento XML muy sencillo, que iremos estudiando para ver las características del lenguaje:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ficha>
<nombre>Felix</nombre>
<apellido>Belzunce</apellido>
<direccion>C/Antonio Oliver, 5</direccion>
</ficha>
```

Lo primero que tenemos que observar es la primera línea. Con ella deben empezar todos los documentos XML, ya que es la que indica que lo que la sigue es XML. Aunque es opcional, es más que recomendable incluirla siempre. Puede tener varios atributos (los campos que van dentro de la declaración), algunos obligatorios y otros no:

- version: indica la versión de XML usada en el documento. La actual es la versión 1.0, con lo que no debe haber mucho problema. Es obligatorio ponerlo, a no ser que sea un documento externo a otro que ya lo incluía (ya veremos qué documentos externos puede haber).
- encoding: la forma en que se ha codificado el documento. Se puede poner cualquiera, y depende del parser el entender o no la codificación. Por defecto es UTF-8, aunque podrían ponerse otras, como UTF-16, US-ASCII, ISO-8859-1, etc. No es obligatorio salvo que sea un documento externo a otro principal.
- standalone: indica si el documento va acompañado de un DTD ("no"), o no lo necesita ("yes"); en principio no hay porqué ponerlo, porque luego se indica en el DTD si se necesita.

4.3.1 Sintaxis de un documento XML

Antes de entrar en el estudio de las etiquetas, hay que resaltar algunos detalles importantes y a los que nos debemos acostumbrar:

- Los documentos XML son sensibles a mayúsculas, esto es, en ellos se

diferencia las mayúsculas de las minúsculas. Por ello <FICHA> sería una etiqueta diferente a <ficha>.

- Además todos los espacios y retornos de carro se tienen en cuenta (dentro de las etiquetas, en los elementos).
- Hay algunos caracteres especiales reservados, que forman parte de la sintáxis de XML: <, >, &, " y '. En su lugar cuando queramos representarlos deberemos usar las entidades <, >, &, " y ' respectivamente. Más adelante hablaré de las entidades y lo que son, pero basta con saber ahora que si escribimos cualquiera de las secuencias anteriores equivaldrá a los correspondientes caracteres citados.
- Los valores de los atributos de todas las etiquetas deben ir siempre entrecomillados. Son válidas las dobles comillas (") y la comilla simple (').

Pasando al contenido en sí, vemos etiquetas que nos recuerdan a HTML, y que contienen los datos. Es importante diferenciar entre elementos y etiquetas: los elementos son las entidades en sí, lo que tiene contenido, mientras que las etiquetas sólo describen a los elementos. Un documento XML está compuesto por elementos, y en su sintáxis éstos se nombran mediante etiquetas.

Hay dos tipos de elementos: los vacíos y los no vacíos. Hay varias consideraciones importantes a tener en cuenta al respecto:

- Toda etiqueta no vacía debe tener una etiqueta de cerrado: <etiqueta> debe estar seguida de </etiqueta>. Esto se hace para evitar la aberración (en el buen sentido de la palabra) a la que habían llegado todos los navegadores HTML de permitir que las etiquetas no se cerraran, lo que deja los elementos sujetos a posibles errores de interpretación.
- Todos los elementos deben estar perfectamente anidados: no es válido poner:

```
<ficha><nombre>Felix</ficha></nombre>
```

y sí lo es sin embargo:

```
<ficha><nombre>Felix</nombre></ficha>
```

- Los elementos vacíos son aquellos que no tienen contenido dentro del documento. Un ejemplo en HTML son las imágenes. La sintáxis correcta para estos elementos implica que la etiqueta tenga siempre esta forma: <etiqueta/>.

Hasta aquí la sintáxis de XML resumida. Aunque la especificación entera es más prolija en cuanto a detalles sintácticos, codificaciones. Ahora quedan por ver otros aspectos, el más prioritario, los DTD.

4.4 DTD: Definición de Tipos de Documento

Como antes se comentó, los documentos XML pueden ser válidos o bien formados. En cuanto a los válidos, ya sabemos que su gramática está definida en los DTD.

Pues bien, los DTD no son más que definiciones de los elementos que puede incluir un documento XML, de la forma en que deben hacerlo (qué elementos van dentro de otros) y los atributos que se les puede dar. Normalmente la gramática de un lenguaje se define mediante notación EBNF; si alguno la conoce, se habrá dado cuenta de que es bastante engorrosa. Pues el DTD hace lo mismo pero de un modo más intuitivo.

Hay varios modos de referenciar un DTD en un documento XML:

- Incluir dentro del documento una referencia al documento DTD en forma de URI (Universal Resource Identifier, o identificador universal de recursos) y mediante la siguiente sintáxis:

```
<!DOCTYPE ficha SYSTEM
"http://www.felix.host.sk/~ficheros/DTD/ficha
.dtd">
```

  En este caso la palabra SYSTEM indica que el DTD se obtendrá a partir de un elemento externo al documento e indicado por el URI que lo sigue, por supuesto entrecomillado.

- O bien incluir dentro del propio documento el DTD de este modo:

```
<?xml version="1.0"?>
<!DOCTYPE ficha [
        <!ELEMENT ficha (nombre+, apellido+, direccion+,
        foto?)>
        <!ELEMENT nombre (#PCDATA)>
        <!ATTLIST nombre sexo (masculino|femenino)
        #IMPLIED>
        <!ELEMENT apellido (#PCDATA)>
        <!ELEMENT direccion (#PCDATA)>
        <!ELEMENT foto EMPTY>
]>
<ficha>
<nombre>Felix</nombre>
<apellido>Belzunce</apellido>
<direccion>C/Antonio Oliver, 5</direccion>
</ficha>
```

La forma de incluir el DTD directamente como en este ejemplo pasa por añadir a la declaración <!DOCTYPE y después del nombre del nombre del tipo de documento, en vez de la URI del DTD, el propio DTD entre los símbolos '[' y ']'. Todo lo que hay entre ellos será considerado parte del DTD.

En cuanto a la definición de los elementos, es bastante intuitiva: después de la cláusula <!ELEMENT se incluye el nombre del elemento (el que luego se indicara en la etiqueta), y después diferentes cosas en función del elemento:

■ entre paréntesis, si el elemento es no vacío, se indica el contenido que puede tener el elemento: la lista de elementos hijos o que descienden de él si los tiene, separados por comas; o el tipo de contenido, normalmente #PCDATA, que indica datos de tipo texto, que son los más habituales.

■ si es un elemento vacío, se indica con la palabra EMPTY.

Ejemplos de cada caso se pueden ver en el DTD de muestra.

A la hora de indicar los elementos descendientes (los que están entre paréntesis) vemos que van seguidos de unos caracteres especiales: '+', '*', '?' y '|'. Sirven para indicar qué tipo de uso se permite hacer de esos elementos dentro del documento:

- + : uso obligatorio y múltiple; permite uno o más elementos de ese tipo dentro del elemento padre, pero como mínimo uno.
- * : opcional y múltiple; puede no haber ninguna ocurrencia, una o varias.
- ? : opcional pero singular; puede no haber ninguno o como mucho uno.
- | : equivale a un OR, es decir, da la opción de usar un elemento de entre los que forman la expresión, y solo uno.

De este modo, si por ejemplo encontramos en un DTD la siguiente declaración:

```
<!ELEMENT ficha (nombre+, apellido+,
direccion*, foto?, telefono*|fax*)>
```

sabremos del elemento ficha que puede contener los siguientes elementos: un nombre y un apellido como mínimo, pero puede tener más de uno de cada; opcionalmente puede incluirse una o varias direcciones, pero no es obligatorio; opcionalmente también se pu ede incluir una única foto; y por fin, pueden incluirse, aunque no es obligatorio en ninguno de los dos casos, uno o más teléfonos o uno o más números de fax.

Como ya se comentó un documento XML presenta una jerarquía muy determinada, definida en el DTD si es un documento válido, pero siempre inherente al documento en cualquier caso (siempre se puede inferir esa estructura a partir del documento sin necesidad de tener un DTD en el que basarse), con lo que se puede representar como un árbol de elementos. Existe un elemento raíz, que siempre debe ser único (sea nuestro documento válido o sólo bien formado) y que se llamará como el nombre que se ponga en la definición del <!DOCTYPE si está asociado a un DTD o cualquiera que se dese e en caso contrario. Y de él descienden las ramas de sus respectivos elementos descendientes o hijos. De este modo, la representación en forma de árbol de nuestro documento XML de ejemplo sería:

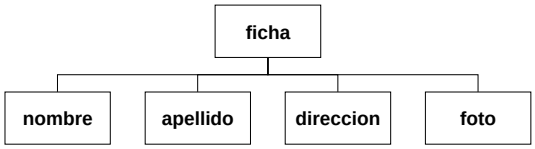


Figura 4.1 Esquema de ejemplo de ficha XML

Vemos que es un documento muy sencillo, con una profundidad de 2 niveles nada más: el elemento raíz ficha, y sus hijos nombre, apellido, dirección, foto. Es obvio que cuanta más profundidad, mayor tiempo se tarda en procesar el árbol, pero la dificultad siempre será la misma gracias a que se usan como en todas las estructuras de árbol algoritmos recursivos para tratar los elementos (comentario para aquellos que han programado algo con árboles).

El DTD, por ser precisamente la definición de esa jerarquía, describe precisamente la forma de ese árbol. La diferencia (y la clave) está en que el DTD define la forma del árbol de elementos, y un documento XML válido puede basarse en ella para estructurarse, aunque no tienen que tener en él todos los elementos, si el DTD no te obliga a ello. Un documento XML bien formado sólo tendrá que tener una estructura jerarquizada, pero sin tener que ajustarse a ningún DTD concreto.

Para la definición de los atributos, se usa la declaración <!ATTLIST, seguida de:

- El nombre de elemento del que estamos declarando los atributos
- El nombre del atributo
- Los posibles valores del atributo, entre paréntesis y separados por el carácter |, que al igual que para los elementos, significa que el atributo puede tener uno y sólo uno de los valores incluidos entre paréntesis. O bien, si no hay valores de finidos, se escribe CDATA para indicar que puede ser cualquier valor (alfanumérico, vamos). También podemos indicar con la declaración ID que el valor alfanumérico que se le de será único en el documento, y se podrá referenciar ese elemento a través de es e atributo y valor
- De forma opcional y entrecomillado, un valor por defecto del atributo si no se incluye otro en la declaración
- Por último, si es obligatorio cada vez que se usa el elemento en cuestión declarar este atributo, es necesario declararlo con la clausula #REQUIRED; si no lo es, se debe poner #IMPLIED, o bien #FIXED si el valor de dicho atributo se debe mantener fijo a lo largo de todo el documento para todos los elementos del mismo tipo (notar que no es lo mismo esto a lo que significaba ID).

4.5 Las Entidades o Entities

Mediante estos elementos especiales es posible dotar de modularidad a nuestros documentos XML. Se pueden definir, del mismo modo que los propios DTD's de los que ya hemos hablado, dentro del mismo documento XML o en DTD's externos.

Las primeras entidades que hemos conocido son los carácteres especiales &, ", ', < y >, que vimos que debíamos escribir mediante las declaraciones: &amp, ", ', < y >. Es decir, que cuando queramos referenciar alguna entidad definida dentro de nuestro documento o en otro documento externo, deberemos usar la sintáxis: &nombre;.

Pero por supuesto las entidades no solo sirven para incluir caracteres especiales no ASCII. También las podemos usar para incluir cualquier documento u objeto externo a nuestro propio documento.

Por ejemplo, y como uso más simple, podemos crear en un DTD o en el documento XML una entidad que referencie un nombre largo:

```
<!ENTITY DAT "Delegación de Alumnos de
Teleco">
```

Y de este modo, cada vez que queramos que en nuestro documento aparezca el nombre "Delegación de Alumnos de Teleco", bastará con escribir &DAT;. Los beneficios son claros, y además es muy sencillo.

El texto referenciado mediante la entidad puede ser de cualquier tamaño y contenido. Si queremos incluir una porción de otro documento muy largo que hemos guradado aparte, de tipo XML o cualquier otro, podemos hacerlo por lo tanto de este modo:

```
<!ENTITY midoc SYSTEM
"http://www.upm.es/~abarbero/midoc.xml">
```

Del mismo modo que usábamos con los DTDs externos, indicamos al procesador con SYSTEM que la referencia es externa y que lo que sigue es una URI estándar, y es lo que queda entrecomillado a continuación.

Hay que resaltar que esto se aplica dentro de documentos XML, pero no de DTDs. Para incluir entidades en DTDs debemos usar el carácter %:

```
<!ENTITY % uno "(uno | dos | tres)">
```

Y para expandirlo en un DTD tendremos que escribir:

```
<!ELEMENT numero (%uno;) #IMPLIED>
```

Capítulo 5

Descripción de las herramientas utilizadas

Dedicaremos este capítulo a explicar las herramientas utilizadas para la realización de este proyecto:

- Familiar Linux
- Orcad Cis Capture
- Orcad Layout
- Compilador Cruzado

Haremos una descripción de cada una de ellas, y explicaremos como instalarlas y utilizarlas.

5.1 Familiar Linux

Familiar Linux es la distribución GNU/Linux diseñada para dispositivos iPAQ, inspirándose en Debian y distribuido en dos versiones distintas, la versión bootstrap en modo consola para desarrolladores y usuarios veteranos, mientras que la segunda versión se basa en los entornos gráficos OPIE o GPE.

El hecho de que se inspire directamente de Debian hace posible la descarga de binarios de un modo similar, gracias a un gestor de paquetes llamado IPKG, el homologo del apt en Debian. Así, podremos instalar, desinstalar, listar paquetes y acceder a los llamados feeds del mundo GNU/Linux y PDA. Se puede utilizar la

herramienta ipkgfind para encontrar paquetes de aplicaciones listas para instalar en la distribución embebida Familiar Linux.

5.1.1 Instalación de Familiar Linux en iPAQ 5500

En primer lugar se descarga el fichero necesario para la instalación en la url http://familiar.handhelds.org/releases/v0.8.2/install/download.html. En esta página deberemos de seleccionar la versión de iPAQ que tenemos y posteriormente que versión de familiar queremos: la bootstrap, OPI o GPE. En nuestro caso elegiremos OPIE, ya que es una versión con entorno gráfico (a diferencia de la bootstrap) y además que corre mas rápido que la GPE en la iPAQ que estamos utilizando.

En primer lugar transferiremos el BootBlaster y los archivos bootldr que vienen con la distribución de Linux sobre la iPAQ usando uno de los métodos que existen para hacerlo, seguidamente salvaremos una copia del Win Ce que lleva por defecto la máquina y finalmente re-flasharemos la máquina con el CRL bootldr.

Los métodos para la transferencia de archivos a la iPAQ son:

· Usando ActiveSync
· Usando Synce
· Usando una tarjeta CF/MMC/SD
· Usando el demonio FTP
· Usando Pocket IE

Aquí se explica el método ActiveSync, para ver como se hace con los demás métodos basta con ir a la página de familiar[4] y ver las guías que se añaden para cada tipo de instalación.

5.1.2 Instalación de bootloader usando ActiveSync

· Transferimos BootBlaster and bootldr a la iPAQ usando ActiveSync

1. Si ActiveSync no está instalado en el Windows PC, lo instalaremos a través del CD-ROM que viene con la iPAQ Pocket PC siguiendo las instrucciones que aparecen por pantalla.

2. Copiaremos los archivos BootBlaster y bootldr que vienen con la distribución Familiar de Linux si no están ya copiado, los archivos se llaman BootBlaster3900-2.6.exe y bootldr-pxa-2.20.4.bin

3. Conectaremos la cuna de la iPAQ a la corriente

[4] La página web de familiar está en la url http://familiar.handhelds.org/

4. Conectaremos el conector USB de la cuna al PC

5. Metemos la iPAQ en la cuna

6. Copiamos BootBlaster3900-2.6.exe al directorio por defecto de la iPAQ haciendo clic en Explore y arrastrando los iconos allí. Ignorar cualquier mensaje del tipo “podría necesitar conversión”.

7. Hacer lo mismo con bootldr-pxa-2.20.4.bin

· Arrancar BootBlaster

1. Seleccionar “Start->Programs” sobra la pantalla táctil de la iPAQ
2. Pincha en File Explorer
3. Pincha sobre BootBlaster

· Salvar la imagen del PocketPC para una posible restauración posterior

1. Ejecutar “Flash->Save Bootldr .gz Format” en BootBlaster para salvar el bootloader en el archivo “\My Documents\saved_bootldr.gz” en la iPAQ.

Hay que tener en cuenta que el Bootloader de Linux también inicia el PocketPC, así que la restauración de este archivo generalmente no es necesaria. Si bien es cierto que por el momento existe un bug en el Bootloader de Linux que provoca que el PocketPC se reinicialice cada unos pocos inicios. Por supuesto podríamos querer guardar y restaurar este bootloader si se restaura el PocketPC

2. Ejecutar “Flash-> Save Wince .gz Format” en BootBlaster para salvar la imagen del PocketPC en el archivo “\MyDocuments\wince_image.gz” sobre la iPAQ. Este proceso puede llevar unos dos o tres minutos aproximadamente.

Si por el contrario no deseamos hacer un backup de nuestro sistema en la iPAQ, podemos desechar este paso entero.

Nota: Hay que tener en cuenta que este proceso salva el bootloader y la imagen ejecutable del PocketPC, es decir, no guarda ningún dato que hayamos introducido en la iPAQ. Deberemos sincronizar el PC con la iPAQ para guardar estos datos. Familiar por el momento no posee ningún sincronizador de datos bajo Linux.

· Copiar saved_bootldr.gz y wince_image.gz al PC con Windows

1. Seleccionar “View->Refresh” en el ActiveSync. Los iconos saved_bootldr.gz y wince_image.gz deberían aparecer.

2. Arrastra los dos archivos del paso anterior del explorador de ActiveSync a un directorio local en nuestro PC.

Se recomienda guardar estos dos archivos en un directorio seguro y sobretodo comprobar el checksum o suma de verificación antes de proseguir para asegurarnos que los datos están perfectamente guardados.

· Instalación de bootloader

Antes de continuar debemos de asegurarnos de que la iPAQ está conectada a una fuente de alimentación externa y que la batería está cargada, para protegerla contra pequeños cambios de potencia durante el breve periodo en el que la iPAQ se esta reprogramando. NO pulsar el botón de alimentación ni el de reset hasta que no hayamos realizado el “Verify” del paso de abajo.

Del menú del BootBlaster, seleccionar “Program”

Un cuadro de dialogo se abrirá permitiendonos seleccionar el bootloader que vamos a utilizar. Seleccionar bootldr.bin.gz, que podría tener un número de versión. Se utilizan archivos gzip porque tienen un checksum interno.

Ser pacientes. El proceso de programar el bootloader puede llevar alrededor de 15 segundos. No interrumpir este proceso, o la iPAQ podría quedar en un estado de inutilización.

Sobre el menú “Flash” de BootBlaster, seleccionar “Verify”

1. Si no dice que tienes un bootloader válido, NO resetear la iPAQ y tampoco apagarla

2. Por el contrario, intenta programar la flash de nuevo

3. Si no funciona, programa tu flash con el bootloader salvado

4. Si continua sin funcionar, envía un mail a bootldr@handhelds.org y/o entra en el canal IRC de familiar. Deja la iPAQ conectada a la corriente y no loa apagues ni le hagas un reset.

Si todo ha ido correctamente, has instalado correctamente el programa CRL bootldr, el cual puede arrancar en cualquier Linux o PocketPC. Como todavía la imagen del PocketPC continua intacta deberíamos de resetear de la forma habitual, en el siguiente paso instalaremos Linux.

5.1.3 Instalación de Familiar 0.8.2 a través del puerto serie

Se necesita un cable serie o una cuna con un cable serie. El cable dual de la cuna USB/Serial que viene con la H3800 y H3900 funcionaran correctamente. También se necesitará de un programa terminal como el kermit o minicom de Linux o el Hyperterminal de Windows.

Si utilizas minicom o kermit, necesitarás un programa ymodem como el sb, que esta disponible en el Linux Irzsz package.

1. Pulsa hacia abajo el joypad y pulsa el botón de reset de la iPAQ. Necesitarás quitar la iPAQ de la cuna para acceder al botón de reset.

· Para usuarios que no sean H5xxx: Cuando el bootloader aparece en la pantalla, liberar en joypad.

· Para usuarios con la H5xxx: Cuando la iPAQ vibre, libera el joypad. La pantalla no cambiara de la imagen previamente mostrada. Si la iPAQ no vibra, quita el adaptador de corriente y la batería, reinserta la batería y el conector de corriente e intenta este paso de nuevo.

2. Pulsar el botón del calendario de la iPAQ. Este es el botón de mas a la izquierda, llamado “Serial Bootldr Console” sobre la pantalla.

3. Cerciorate de que el terminal está funcionando, y que interactúa de forma correcta con el bootloader. Una correcta interacción consiste en conseguir mandar comandos y recibir respuestas. La configuración del emulador terminar debe de ser como sigue: 115200 8N1 configuración serie, no control de flujo y no hardware handshaking.

Si no puedes inter accionar con con el bootloader, cerciorate de que la configuración del emulador terminal es correcta, de que la iPAQ esta correctamente conectada al PC y de que está encendida. Si todo parece correcto, intenta resetear el emulador terminal y la iPAQ.

En algunas ocasiones Hyperterminal utiliza el 100% de la CPU sin permitir ninguna interacción con la iPAQ. En este caso necesitaras ir al administrador de tareas para cerrar el Hyperterminal antes de que tu la puedas resetear.

4. En el “boot>” prompt, pone l siguiente comando: load root

5. Proceder a enviar o subir el archivo jffs2 con ymodem, usando el emulador terminal. Si no ha sutilizado ymodem antes o tienes alguna pega mira: handhelds-faq/getting-started.html#USING-XYZMODEM.

Ten en cuenta que el bootldr espera por defecto un ymodem, y no un xmodem como en versiones anteriores. Si tu no puedes utilizar ymodem por alguna razón, tu puedes pasarte a xmodem con el comando set ymodem 0.

Se deberá de ver algo así:

```
boot> load root
loading flash region root
ready for YMODEM download..
Erasing sector 00140000
Erasing sector 00180000
Erasing sector 001C0000
Erasing sector 00200000
.
.
.
addr: 00360000 data: 781590DB
addr: 00370000 data: 642637AE
addr: 00380000 data: E0021985
addr: 00390000 data: 15DA97EC
Erasing sector 00FC0000
writing flash..
addr: 00100000 data: E0021985
addr: 00110000 data: E3BAD617
addr: 00120000 data: 0FA1F57B
addr: 00130000 data: 9343AEEB
.
.
.
addr: 00600000 data: E0021985
addr: 00610000 data: FFFFFFFF
addr: 00620000 data: FFFFFFFF
addr: 00630000 data: FFFFFFFF
verifying ... formatting ... done.
boot>
```

5.2 Orcad 9.0

OrCAD 9.0 es una herramienta informática integral, es decir, que integra muchas herramientas con las que hacer de todo en el mundo de la electrónica. Por ejemplo incluye:

- Pspice: el simulador analógico - digital
- Capture: el esquemático para hacer todos tus circuitos, tanto digitales como analógico
- Layout: te permite hacer "layouts" de placas, para insolarlas después
- Editor: editor de componentes..
- Y algunas herramientas más....

Además de la funcionalidad de todas las herramientas, dispone de un entorno CIS, que permite la intercomunicación entre todos los componentes.

5.2.1 Orcad Cis Capture

Para construir el esquemático del circuito y para la simulación del generador de ondas cuadradas se ha utilizado la herramienta Orcad Cis Capture. Orcad Cis Capture es uno de los esquemáticos más utilizados en diseño electrónico dentro del ámbito profesional y universitario. Además, permite a partir de los esquemáticos que hacemos, preparar toda la información generada para la simulación o la ejecución de circuitos impresos.

Las principales características del Cis Capture son:

· Editor de esquemático rápido e intuitivo
· Fácil organización y rehúso de circuitos duplicados a través bloques jerárquicos
· Crea y edita partes en la librería o directamente de la página del esquemático sin interrumpir la ventana de trabajo
· Auto integración de dispositivos PLD y FPGA en los esquemáticos
· Acceso a algunas o todas las partes, redes, pines y bloque de propiedades, y posibilidad de realizar cambios a través del editor spreadsheet
· Realiza presupuestos en función de los elementos que haya en el esquemático
· Identifica partes visualmente, modifica las propiedades que sean necesarias y las sitúa en un diseño.
· Interfaz con otras aplicaciones Orcad como la layout para acelerar el proceso de diseño.

5.2.2 Orcad Layout

Layout es la solución PCB clásico, con autorouter de 16 capas que permite diseñar la gran mayoría de circuitos impresos.

5.2.3 Instalación de Orcad 9

Orcad corre únicamente en máquinas con el sistema operativo Windows, así que su instalación es la propia de todos los programas que funcionan bajo Windows. Únicamente tendremos que pinchar sobre setup y seguir las instrucciones que parecen en pantalla.

5.3 Compilador Cruzado

Por un lado el programa cliente debe funcionar sobre la PDA iPAQ, mientras que el programa de toma de decisión lo ha de hacer sobre el mini- ordenador desarrollado por la empresa Israelita Compulab. Ambos dispositivos tiene en común que utilizan un procesador que no es habitual en los PC´s de casa, el procesador XSCALE. Son

una nueva generación de microprocesadores sucesores de los StrongARM, propietario de Intel. Estos procesadores están dirigidos fundamentalmente a los equipos móviles que utilizan tecnología inalámbrica para sus comunicaciones.

Debido a que los dispositivos que estamos utilizando poseen un procesador XSCALE necesitamos de un compilador cruzado. Un compilador cruzado es un compilador capaz de crear código ejecutable en otra plataforma distinta a aquélla en la que él se ejecuta. Esta herramienta es útil cuando quiere compilarse código para una plataforma a la que no se tiene acceso, o cuando es incómodo o imposible compilar en dicha plataforma (como en el caso de los sistemas empotrados). El compilador cruzado utilizado en este proyecto ha sido el gcc versión 3.0.

Capítulo 6

Arquitectura e implementación

Arquitectura

6.1 Proceso cliente

6.1.1 Introducción

El proceso cliente intentará constantemente conectarse al proceso servidor para que una vez que se consiga establecer la conexión (el cliente ha entrado dentro de la zona de cobertura del servidor), le envíe una ficha XML al servidor. Dicha ficha contendrá información acerca de la discapacidad o discapacidades del individuo.

El principal problema que surge es cómo el proceso cliente es capaz de conocer que ha salido de la zona de cobertura una vez que ha enviado la ficha, para a partir de entonces intentar conectarse a un nuevo servidor. Éste problema se aborda detenidamente en el apartado del funcionamiento del Keep Alive..

Para todo lo que sigue hay que tener en cuenta que el sistema funciona bajo el estándar 802.11 y que por tanto todas las comunicaciones que se realizan entre cliente y servidor utilizan el aire como medio de transmisión.

6.1.2 Funcionamiento del proceso cliente

En las siguientes figuras se representa de forma simplificada la forma en la que interaccionan cliente y servidor, y cómo es el funcionamiento del cliente mediante un diagrama de flujo.

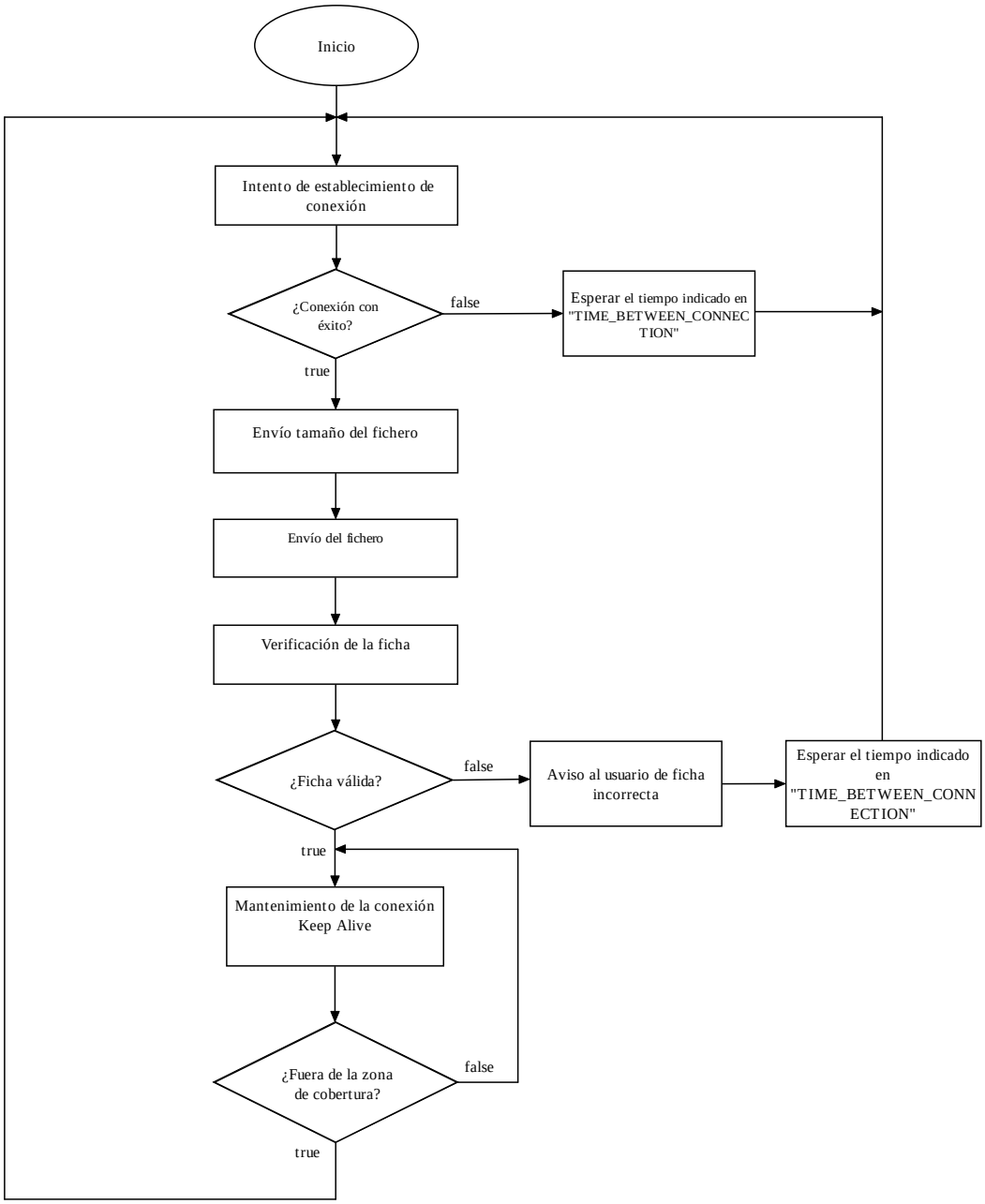


Figura 6.1. Funcionamiento del cliente

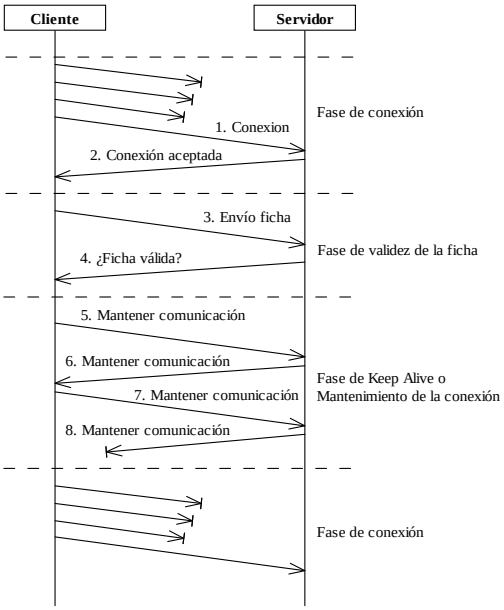


Figura 6.2. Interacción entre cliente y servidor

Como podemos observar en las figuras anteriores, el cliente intentará establecer una conexión con el proceso servidor. Si la conexión no se ha conseguido establecer, el proceso “dormirá” durante un tiempo y volverá a intentar conectarse a un nuevo servidor.

En algún punto de la estructura de directorios del cliente se encuentra una ficha XML donde se describe el tipo o tipos de discapacidades que tiene esa persona. Pues bien, una vez establecida la conexión el cliente deberá enviar el fichero XML al proceso servidor, indicandole antes el tamaño del mismo. Se podría pensar que no es necesario enviar el tamaño del fichero, pero se hace para saber en donde termina el envío del fichero y donde empieza la verificación de la ficha. Las fichas XML no van a tener siempre el mismo tamaño.

Una vez que se ha enviado el fichero el proceso espera a que el servidor confirme que esa ficha es correcta. En caso contrario se informa al cliente a través de un mensaje indicándole que la ficha es incorrecta, pero el funcionamiento del cliente continúa intentando conectarse a un nuevo servidor.

Si todo lo anterior funcionó correctamente se lleva a cabo el mantenimiento de la conexión o Keep alive. Esta función da respuesta a una de las problemáticas del proyecto que consistía en responder a la pregunta: "¿Cómo sabemos que el cliente ha salido de la zona de cobertura?".

6.1.3 Funcionamiento del Keep Alive

A continuación se describe cómo funciona la función Keep Alive. En primer lugar, enviamos un dato y esperamos a que se reciba el mismo dato que habíamos enviado durante un tiempo limitado.

Si salta el time out, cuyo tiempo está definido en una estructura de tiempo, entonces la función Keep Alive termina, lo que indica que el cliente ha salido de la zona de cobertura.

Por el contrario, no salta el time out, si recibimos el mismo dato que habíamos enviado, se determina que la conexión continua y se enviará un dato distinto, volviendose a repetir el proceso anteriormente descrito hasta salir de la zona de cobertura.

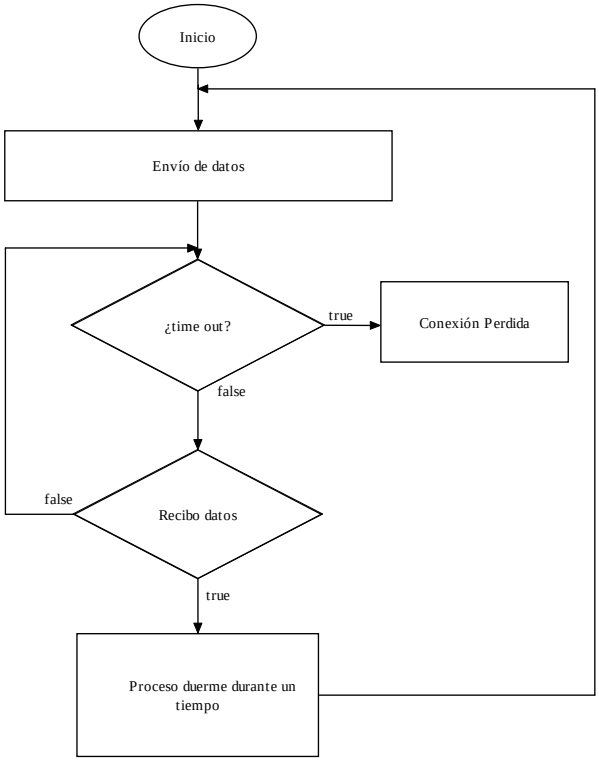


Figura 6.3. Función Keep-Alive

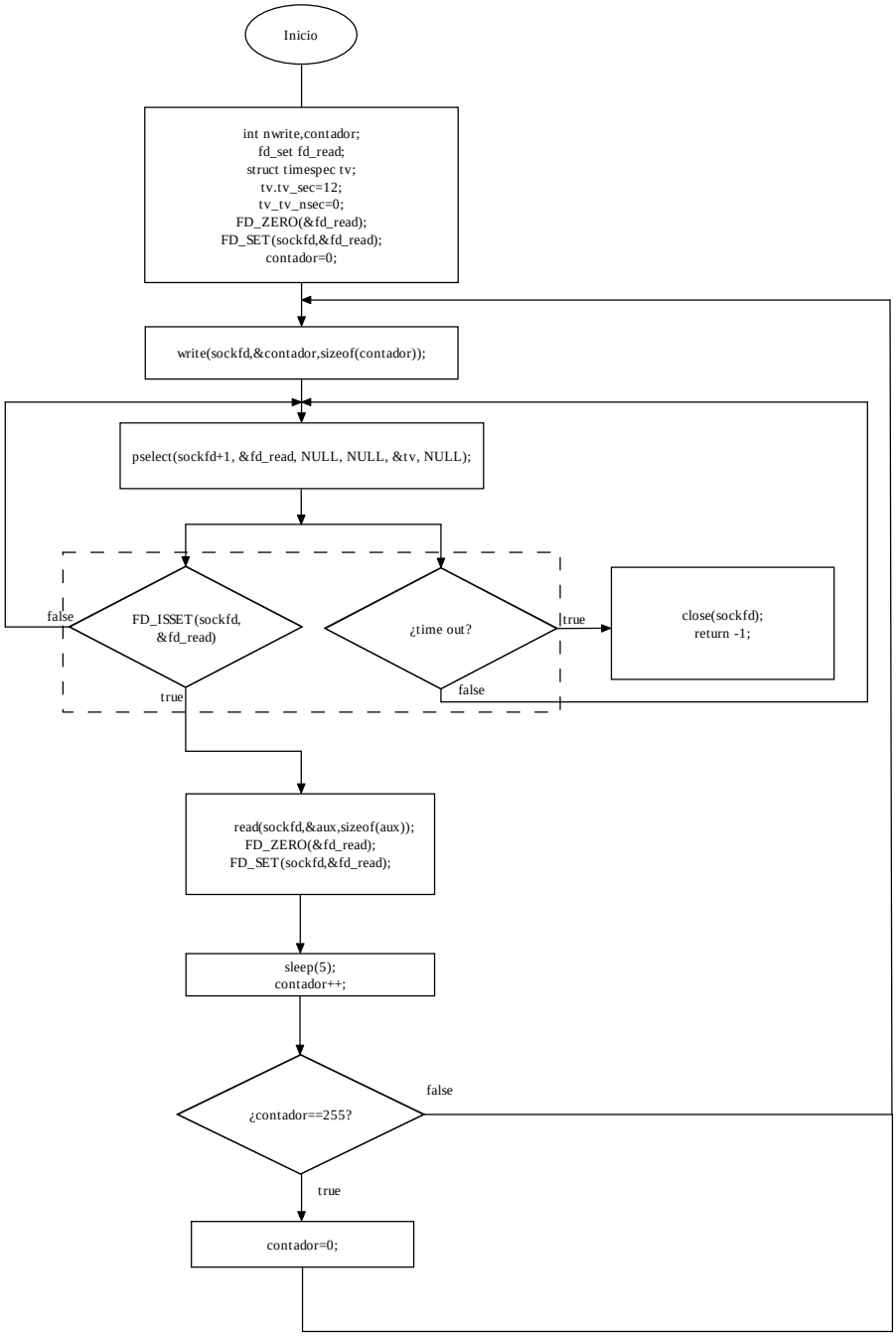


Figura 6.4.Función Keep Alive con código en c

6.2 Módulo servidor de toma de decisión

6.2.1 Introducción

El proceso servidor de toma de decisión es el encargado de controlar una placa que simula un semáforo con generador de tonos audibles en función de la información que le transfiere el módulo procesador XML.

Como podemos observar en la siguiente figura, existen tres procesos que corren de forma local dentro del servidor: el que se encarga de atender las peticiones, el procesador XML y la toma de decisión. También se puede apreciar el sentido de la comunicación y cómo interaccionan entre sí los distintos procesos.

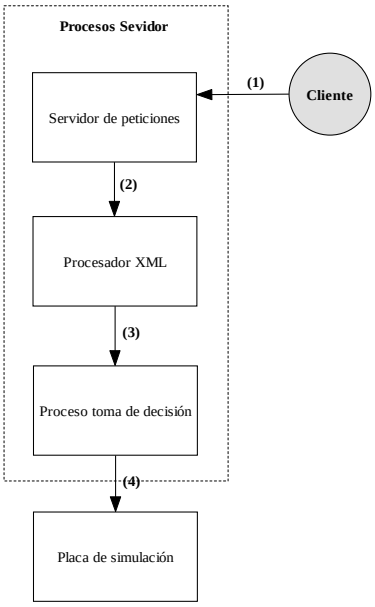


Figura 6.5. Interacción entre los procesos que corren en el servidor

En (1) de la Figura 6.5, un Cliente se conecta al proceso que atiende las peticiones para enviarle una ficha XML, esta ficha pasa de forma local al procesador XML (2). Éste determinará si esa ficha es válida o si por el contrario es incorrecta. Si la ficha es válida el procesador XML transmitirá de forma local el tipo de discapacidad que tiene el individuo al proceso toma de decisión (3) para que éste ya realice un control adecuado sobre la placa (4).

Cabe destacar que en este primer diseño sólo se han tenido en cuenta dos tipos de discapacidades: discapacidad visual y locomotriz, y tres grados de deficiencia dentro de cada tipo: grado uno, dos y tres. Por tanto, el proceso toma de decisión sólo sabrá actuar sobre este tipo de discapacidades, cualquier otra

discapacidad no tendrá ningún efecto a la hora de controlar la placa de simulación.

6.2.2 Funcionamiento del proceso toma de decisión

El proceso de toma de decisión a través del control del puerto paralelo hace que los 3 led`s de la placa que simulan un semáforo vayan pasando por cada uno de los tres estados (verde, ámbar y rojo) continuamente.

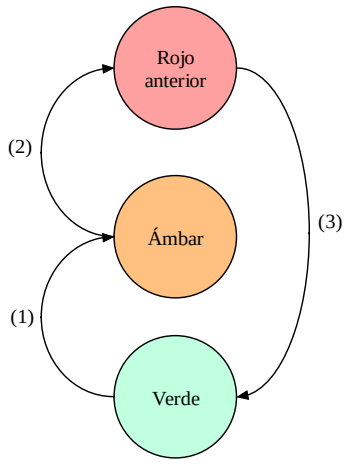


Figura 6.6. Semáforo

Transición entre estados y acceso a toma de decisión

La siguiente figura nos muestra cuándo y cómo se produce la transición entre estados y se accede a toma de decisión, que es la función encargada de realizar variaciones sobre las temporizaciones iniciales de cada uno de los estados y de activar o no el zumbador del semáforo.

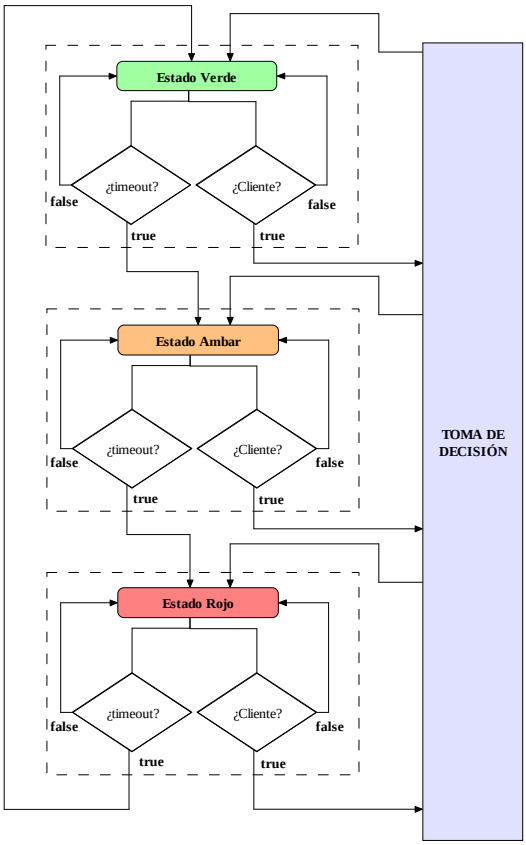


Figura 6.7. Transición entre estados y acceso a toma de decisión

Como podemos observar en el gráfico la transición entre estados se produce cuando se alcanza el timeout que hay en cada uno de los estados, este tiempo viene marcado inicialmente por las constantes que se indican en la figura 6.9. Pero éste puede sufrir modificaciones si en algún momento, uno o varios clientes entran en la zona de cobertura, produciéndose entonces una variación en el funcionamiento normal del semáforo para adapatarlo a las necesidades de los discapacitados.

Según lo visto anteriormente, cuando un cliente entra dentro de la zona de cobertura se accede a toma de decisión independientemente del estado en el que esté. Esto es muy importante porque de esta forma el sistema durante su funcionamiento es capaz de recibir peticiones de los discapacitados independientemente del estado en el que lleguen.

Funcionamiento de toma de decisión

En toma de decisión el funcionamiento del semáforo con zumbador se adaptará a los individuos que hayan llegado al sistema, la adaptación como era de esperar, consiste en la variación de la temporización del estado rojo y la activación o desactivación del zumbador en dicho estado. El entorno se adaptará a los

discapacitados que estén dentro de la zona de cobertura durante los estados rojo, verde y ámbar anteriores al rojo en el que estamos. La figura 6.8 intenta ilustrar precisamente esto último.

Si definimos un ciclo de semáforo como la transición de rojo a verde y de éste a ámbar, los clientes que lleguen en cualquier estado de un ciclo, intervendrán en el rojo del siguiente ciclo.

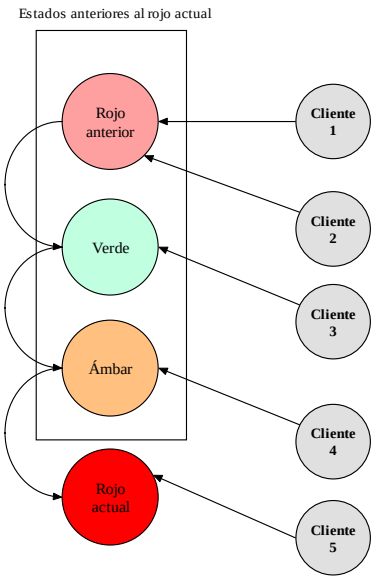


Figura 6.8. Transición de estados en el semáforo

Hemos comentado que toma de decisión hacer variar la temporización del estado rojo y activa o desactiva el zumbador del semáforo dependiendo si las discapacidades que hay en ese momento en el sistema lo requieren. Pero no se ha especificado cómo funciona toma de decisión, es decir, cómo a partir de la información que le pasa el procesador XML es capaz de detectar las discapacidades del individuo, cómo se consigue variar la temporización del rojo y activar o no el zumbador.

La siguiente figura muestra las constantes que definen los tipos de discapacidades así como las temporizaciones en funcionamiento normal y de cada una de las discapacidades.

RESUMEN DE CONSTANTES EN EL PROCESO TOMA DE DECISIÓN

```
/*Tiempos en segundos de cada una de las deficiencias
para la temporización en estado rojo*/

#define DEF_VISUAL_GRADO1 15
#define DEF_VISUAL_GRADO2 10
#define DEF_VISUAL_GRADO3 7

#define DEF_LOCOMOTRIZ_GRADO1 15
#define DEF_LOCOMOTRIZ_GRADO2 10
#define DEF_LOCOMOTRIZ_GRADO3 7

/*Tiempo en segundos para cada uno
de los estados en funcionamiento normal*/

#define TIEMPO_LED_VERDE 7
#define TIEMPO_LED_AMBAR 7
#define TIEMPO_LED_ROJO 7
```

```
/*Numeración de cada una de las deficiencias en
potencias de dos*/

#define DVG1 1
#define DVG2 2
#define DVG3 4
#define DLG1 8
#define DLG2 16
#define DLG3 32
```

Figura 6.9. Resumen de constantes en el proceso toma de decisión

Como podemos observar a las discapacidades hay que asignarles números enteros que son potencias de dos, y en el caso de que una persona tenga varias discapacidades habrá que sumar el número de cada una de las discapacidades que tiene asociadas ese individuo. Esto se hace para poder identificar las discapacidades que tiene una persona realizando una operación lógica “and” entre el número que nos pasa el procesador XML (que indica la discapacidad o las discapacidades) y cada una de las deficiencias que se han definido.

Una vez identificada las discapacidades deberemos comprobar por un lado si hace falta activar el zumbador para el siguiente rojo y por otro lado si la temporización del siguiente rojo que está definida en el sistema es menor que la que nos ha llegado, en cuyo caso habrá que actualizar.

El siguiente esquema trata de presentar de forma visual lo anteriormente expuesto. La variable deficiencia es la que contiene el número que nos pasa el procesador XML , mientras que el segundo parámetro de la función es un puntero a la temporización del siguiente rojo.

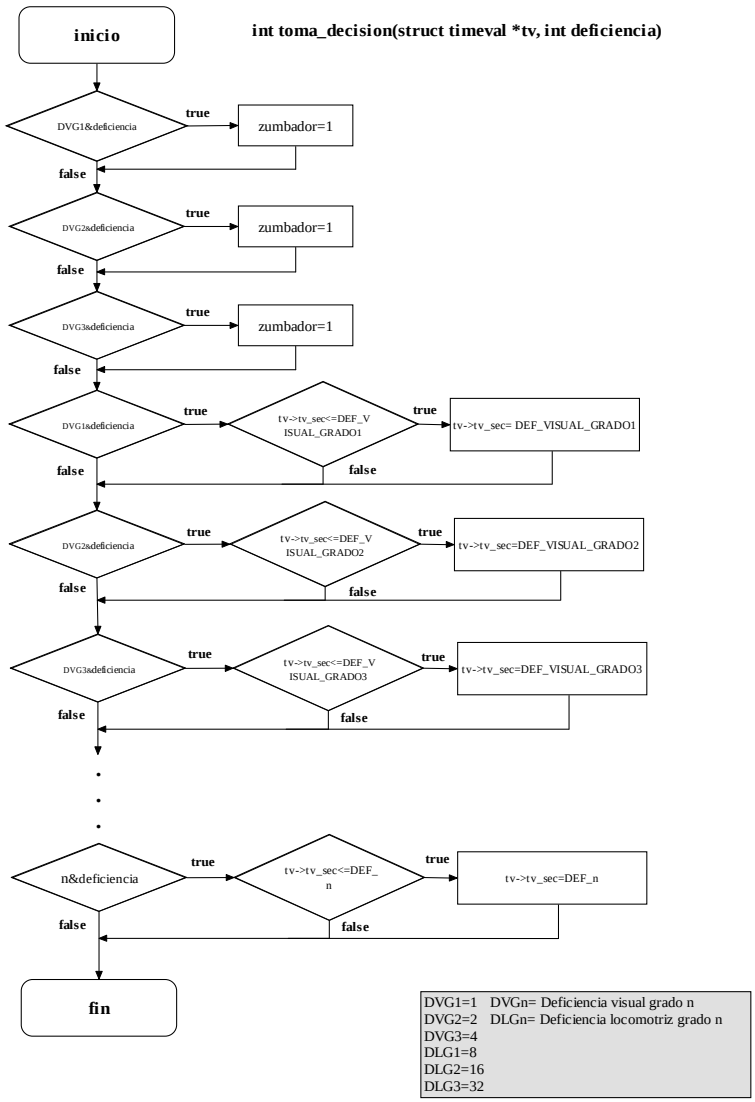


Figura 6.10.Funcionamiento de la función toma de decisión

Ejemplos de como funciona el sistema

Para explicar como se comporta el sistema ante la llegada de clientes vamos a definir tres tipos de situaciones: situación de funcionamiento normal (sin clientes), cliente llegan en verde o ámbar, cliente que llegan en todos los estados.

En todos los gráficos observamos 3 tipos de cuadros: uno en el que se especifican los tiempos que tendrá el semáforo en funcionamiento normal, otro en el que se indican los tipos de discapacidades y el tipo de operación a realizar y por último, se especifica cuales serán las acciones en el siguiente rojo.

En esta primera figura observamos que en los estados verde y ámbar no han existido clientes que hayan entrado dentro de la zona de cobertura. Por tanto, cuando

se produzca la transición de ámbar a rojo, las acciones sobre la temporización del rojo y sobre el zumbador será las propias del funcionamiento normal del sistema

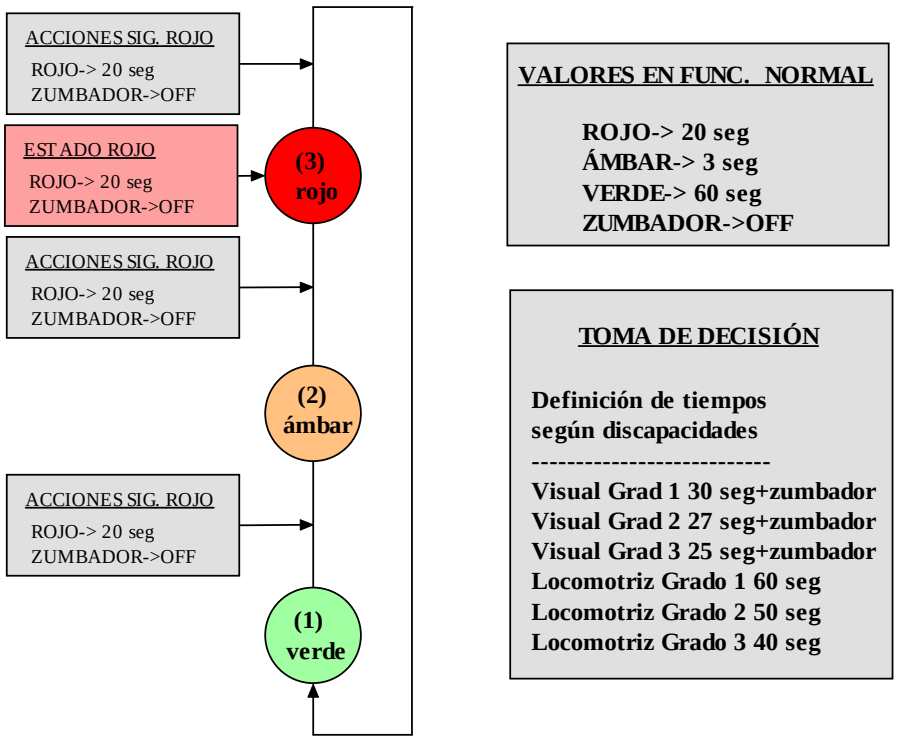


Figura 6.11. Situación de funcionamiento normal

Por el contrario en esta segunda figura podemos observar cómo se produce la llegada de dos clientes en estado verde: uno con deficiencia locomotriz grado uno y el otro con deficiencia visual grado tres, mientras que en estado ámbar solo accede al sistema un discapacitado con deficiencia visual grado uno.

Al llegar el primer cliente las acciones para el siguiente rojo se modifican en función de la temporización que tiene asociado el rojo y si se requiere activar o no el zumbador para este tipo de discapacidad. En este caso la temporización es de 25 segundos y se requiere la activación del zumbador. Posteriormente llega el cliente número dos que tiene una temporización mas restrictiva para el estado rojo, así que se producirá una actualización a esta nueva temporización (60 segundos), como se indica en la figura. Al llegar el tercer cliente no se produce actualización alguna en el sistema, puesto que por un lado el zumbador ya estaba activado y por el otro la temporización del rojo es menor que la que ya existía en el sistema.

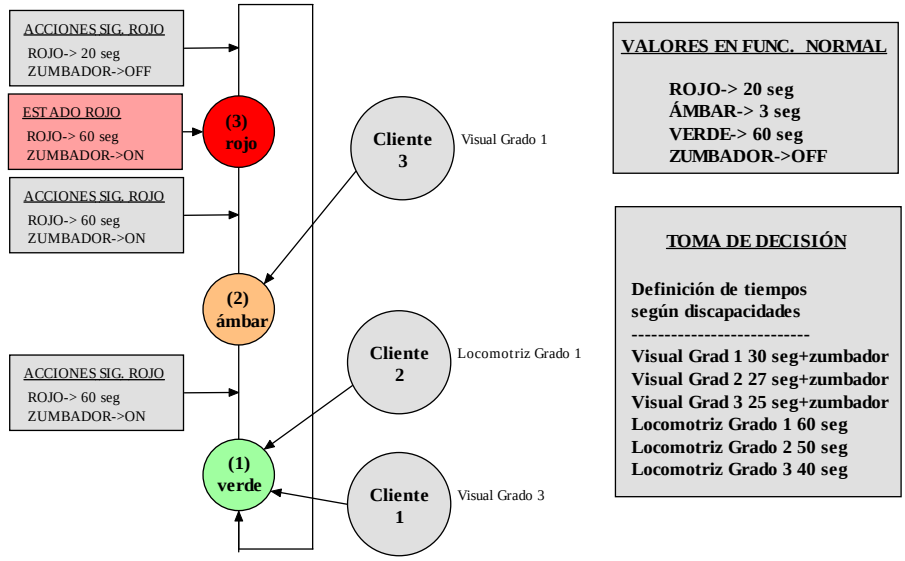


Figura 6.12.Cliente llegan en verde o ámbar

Por último se ilustra una imagen en la que llegan peticiones en todos los estados. El comportamiento del sistema es el de siempre. En este caso comenzamos en estado rojo, que posee una temporización correspondiente al funcionamiento normal del sistema. En ese mismo estado llegan dos cliente con una deficiencia visual de distinto grado, el más restrictivo es el cliente número uno así que el sistema adopta las características asociadas de esta deficiencia. Posteriormente se pasa por estado verde en el que no llega ningún cliente, y por ámbar donde llega un cliente con una deficiencia locomotriz grado tres que tiene una temporización asociada al rojo más grande que la que existe actualmente en el sistema, así que se procede a actualizar la variable de tiempo del siguiente rojo (40 segundos). Por último en (4) observamos que la temporización que se lleva a cabo es la mas restrictiva de todos los clientes que llegaron entre el último rojo y el ámbar, y que por supuesto el funcionamiento del zumbador está activo puesto que llegaron clientes con deficiencia visual.

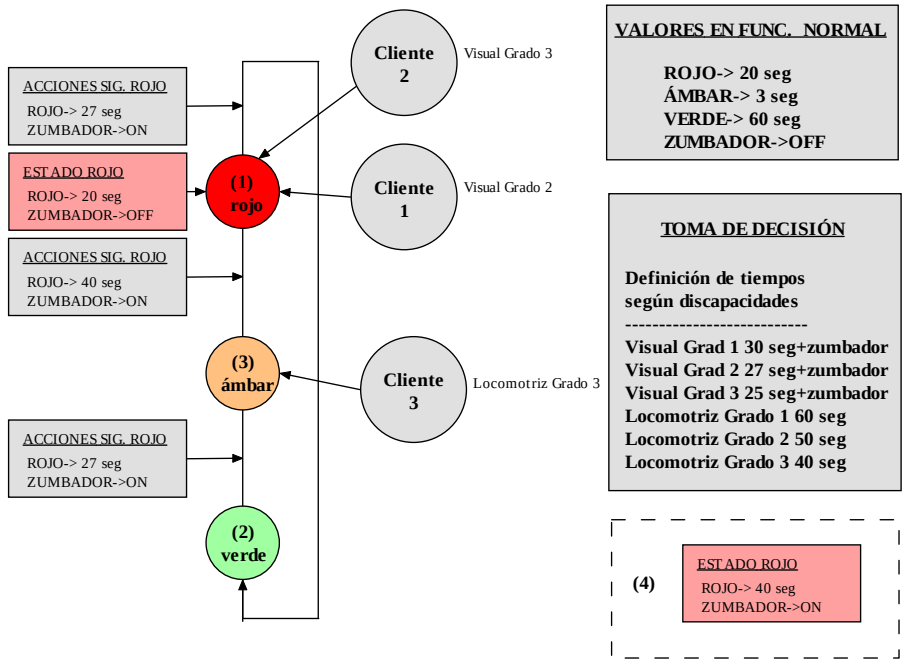


Figura 6.13. Llegada de clientes en todos los estados

6.3 Interfaz de simulación del sistema a través del puerto paralelo

6.3.1 Introducción

Para la simulación del sistema se ha diseñado e implementado un circuito que nos permite simular el funcionamiento de un semáforo con la incorporación de un generador de tonos audibles. Dicho semáforo se conectara al puerto paralelo de un mini-ordenador que será el que lo controle.

6.3.2 Descripción Física de la placa

6.3.2.1 Generador de tonos audibles

El generador de tonos se ha implementado utilizando el circuito integrado 555 con la configuración de multivibrador astable.

En primer lugar comentaremos por encima el funcionamiento interno del circuito integrado 555 para posteriormente observar el funcionamiento del mismo como multivibrador astable, que es la configuración que se ha utilizado en la placa de simulación.

6.3.2.1.1 Funcionamiento interno del circuito integrado 555

El esquema simplificado está representado en la figura 6.14. Como se ve dispone de dos A.O. trabajando como comparadores de tensión, un biestable RS que emplea la salida complementaria Q, dos transistores en conmutación T_1 y T_2, un buffer inversor para proporcionar la alta corriente de salida, y finalmente una red divisora de tres resistencias de valor R.

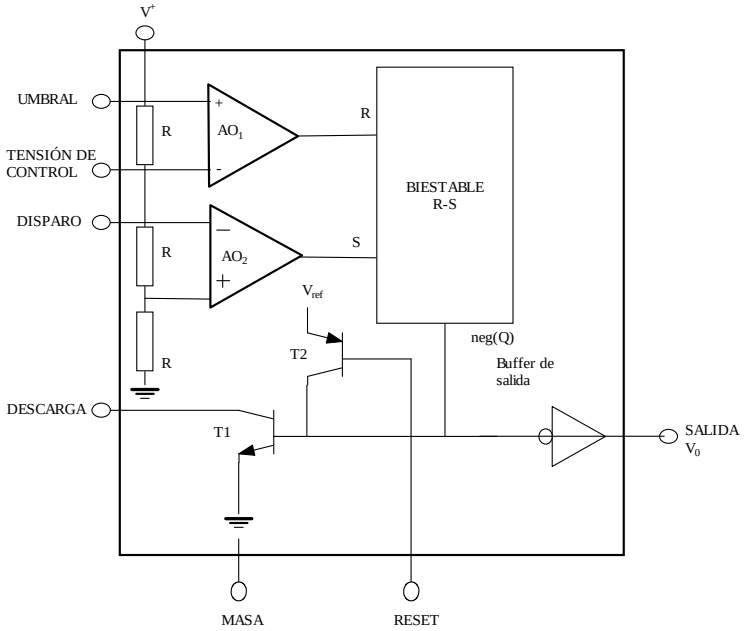


Figura 6.14. Configuración interna del CI 555

Puesto que estas resistencias son iguales, las tensiones de referencia aplicadas a los terminales de entrada de los comparadores de tensión son:

- Para el A.O.1: $V_{(-)}$ = 2/3 V^+;

- Para el A.O.2: $V_{(+)}$ = 1/3 V^+;

Considerando que el circuito está alimentado con polaridad simple, los dos posibles estados de la salida de los comparadores son:

- Nivel “1"= V^+.
- Nivel “0"= 0 V.

Si la tensión en el terminal (+) del A.O.1, UMBRAL, supera los 2/ 3 V^+, su salida pasa a “1", este nivel es la entrada R del biestable, por lo que $\overline{Q}$=”1", de esta manera el transistor T_1 se satura y la salida pasa a “0".

Por el contrario, si la tensión a la entrada (-) del A.O.2, DISPARO, cae por debajo 1/3 V^+, S=”1", lo que implica que $\overline{Q}$=”0", el transistor T_1 se bloquea y la salida pasa a “1".

En cualquier momento podemos “poner a cero” la salida del temporizador, aplicando un “0" en RESET saturamos el transistor T_2, lo que lleva al bloqueo a T_1 y a V_0=”0".

6.3.2.1.2 Circuito integrado 555 como multivibrador astable

Esta es la configuración que utilizaremos en la placa que simula el funcionamiento del sistema para conseguir una onda cuadrada capaz de generar un tono en un altavoz convencional de 8 ohmios. El esquemático del circuito se muestra en la figura 6.16 mientras que la onda que genera la salida está representada en la figura 6.15.

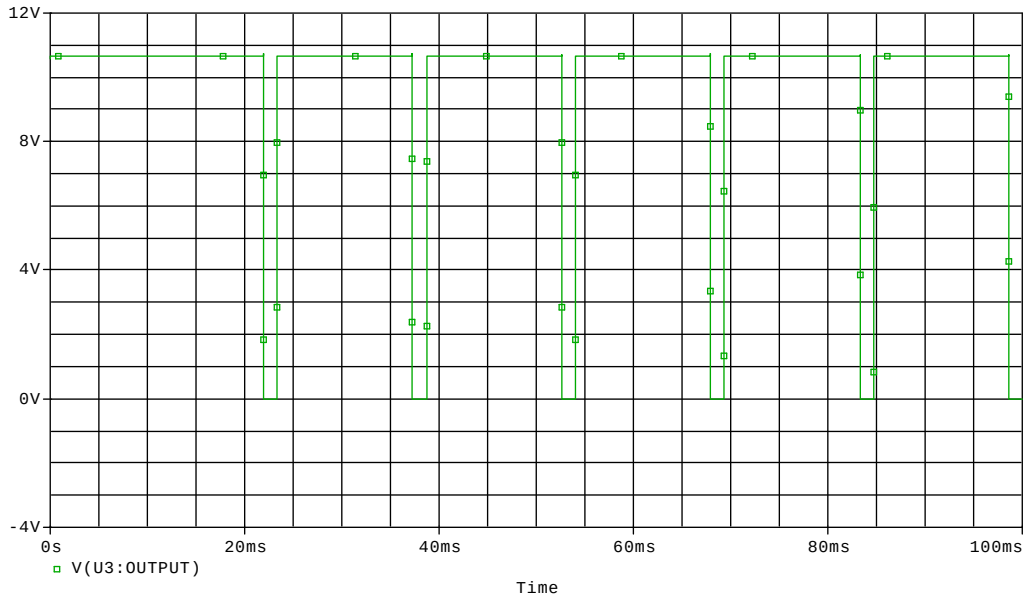


Figura 6.15. Onda generada por el CI 555 como multivibrador astable

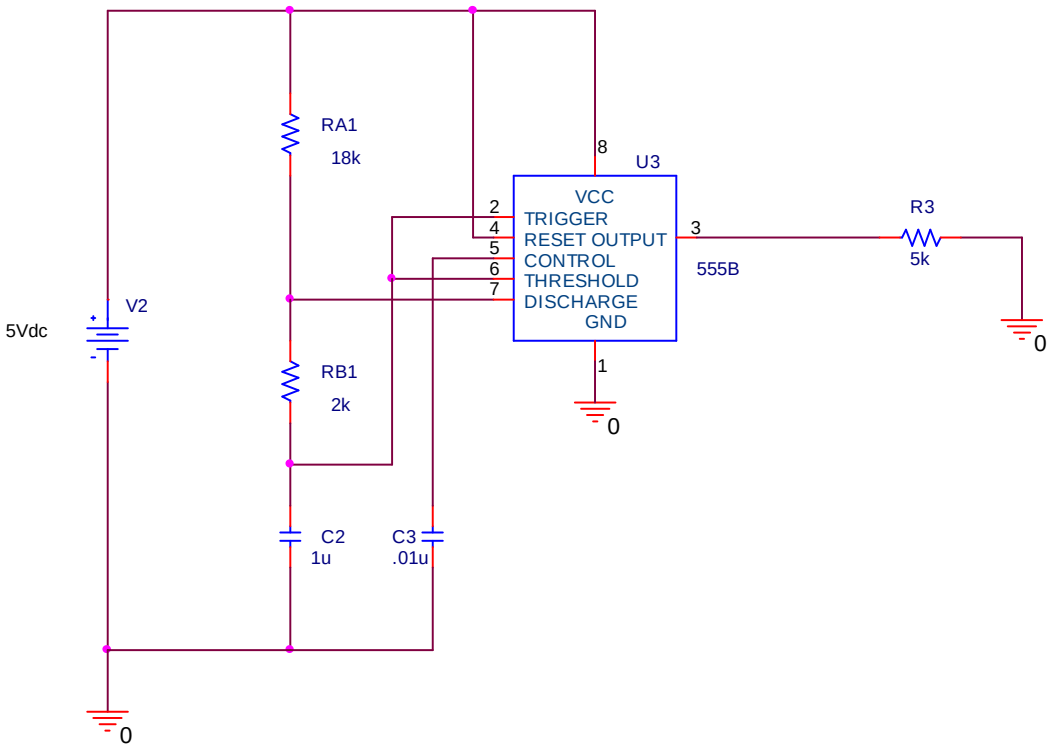


Figura 6.16.Esquemático del CI 555 como multivibrador astable

Al poner en marcha el circuito, el condensador C2 se encuentra descargado, y aplica una tensión inicial cero al terminal UMBRAL, patilla 6, y al terminal DISPARO, patilla2. En estas condiciones R="0" y S="1", por lo que la salida del biestable RS es $\overline{Q}$ ="0", lo que lleva la salida a nivel alto, Vo="1". Esta situación se mantiene mientras dura la carga del condensador C2, a través de las resistencias RA1 y RB1, cuando la tensión en C2 alcanza el valor 2/3 V, la salida pasa a $\overline{Q}$="1", haciendo cambiar la salida a estado bajo, Vo=0 V, e iniciando la descarga de C2 a través de RB1, y del transistor T1, patilla 7. Cuando la tensión en C2 ha decrecido hasta el valor 1/3 V, vuelve a repetirse el ciclo de carga del condensador C2.

Llamando T1 al tiempo en el que la salida permanece en estado alto, o lo que es lo mismo, al tiempo de carga del condensador C, resultará que **T1=0.693*(RA+RB)*C1** y si T2 es el tiempo en estado bajo de la salida, o también el tiempo de descarga del condensador C1, entonces **T2=0.693*RB*C1**.

El periodo total T vendrá dado por:

T=T1+T2=0.693*(RA1+2*RB1)*C2

Y la frecuencia de oscilación es entonces:

F=1/T=1.44/(RA1+2*RB1)*C2

Como caso particular, si hacemos R2>>R1, la fórmula aproximada es:

F=0.72/(RB1*C2)

Esta situación corresponde precisamente a un generador de onda cuadrada simétrica, es decir T1 = T2.

6.3.2.2 Circuito final impreso en placa

Por un lado en la figura 6.17 podemos apreciar que para el diseño del semáforo se han utilizado 3 diodos led´s controlados por los pines 5,6 y 7 del puerto paralelo.

Por otro lado, vemos que el generador de tonos audibles tiene el mismo diseño que el multivibrador astable explicado anteriormente, pero con la diferencia de que a la salida tenemos la presencia del condensador C3. Este condensador sirve para hacer sonar más fuerte el tono en el altavoz.

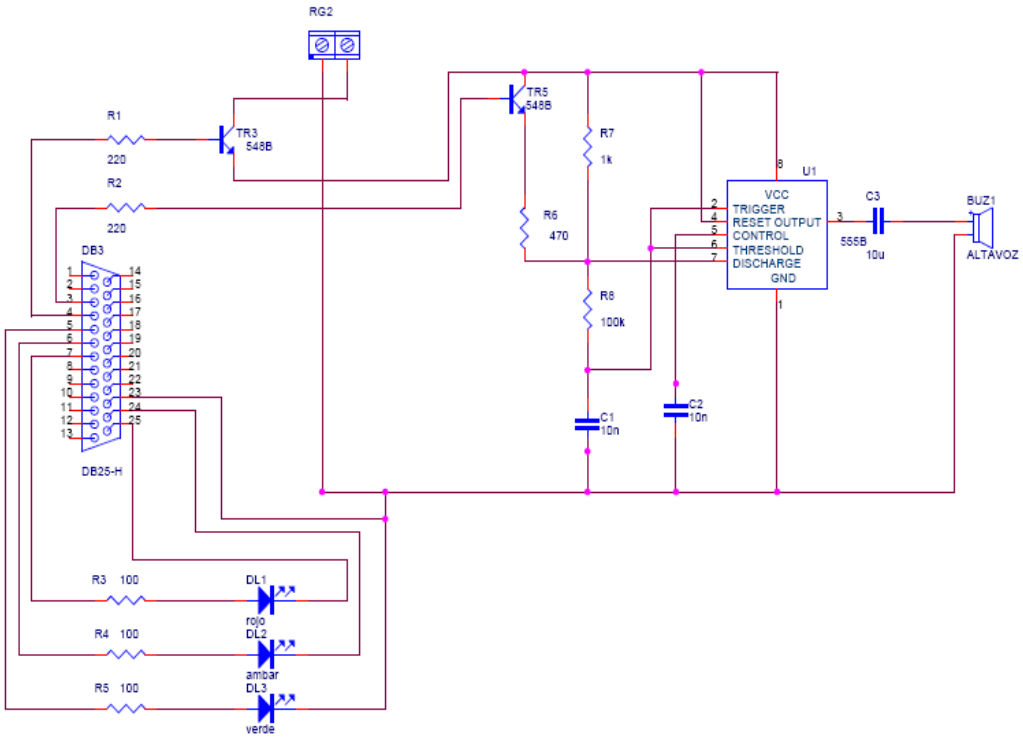


Figura 6.17. Esquemático del circuito impreso en placa

Por otra parte se observa la existencia de 2 transistores TR3 y TR5,. A través del transistor bipolar npn TR5 conseguimos una baja o alta frecuencia en el tono. Así pues, si ponemos a estado alto el transistor TR5, éste se comportará como un interruptor cerrado poniendo en paralelo las resistencias R4 y R5 y provocando un tono de alta frecuencia (high). Por el contrario, si la base del transistor se encuentra

en un estado bajo, el transistor funcionará en un estado de baja frecuencia (low).

Finalmente, con el transistor bipolar npn TR3 permitimos encender o apagar el generador tonos. Un estado alto en la base produce el funcionamiento del generador, mientras que un estado bajo provoca la desconexión del mismo.

La siguiente tabla intenta ilustrar de forma rápida que significado tienen los principales elementos que configuran el circuito.

Elemento/s	**Significado**
Transistor TR3	Encendido o apagado del generador de ondas.
Transistor TR5	Estado de alta o baja frecuencia
LED 1	Luz roja del semáforo
LED 2	Luz ámbar del semáforo
LED 3	Luz verde del semáforo
LED 4	Luz de funcionamiento del generador de ondas
R1,R2 y C1	Frecuencia del tono en estado de bajo
R5,R1,R2 y C1	Frecuencia del tono en estado alto

En las figuras 6.18 y 6.19 , podemos ver como queda el “layout” del circuito. Se puede apreciar que la densidad de pistas es muy baja, es decir, que la placa se podría haber hecho más pequeña, pero la colocación de un porta altavoz, junto con el tamaño del puerto paralelo, han limitado el tamaño del circuito. Además, se pretendía que el tamaño del circuito no fuera demasiado pequeño ya que era un circuito de simulación.

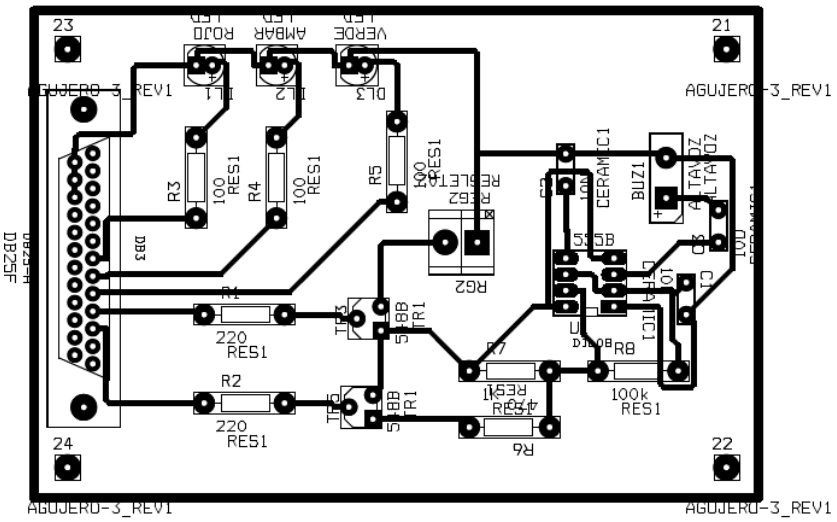


Figura 6.18. “Layout” del circuito.

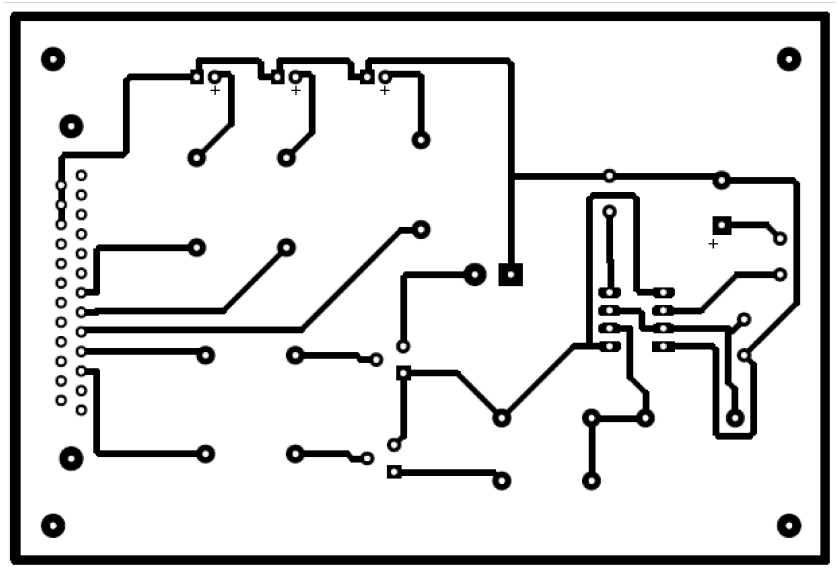
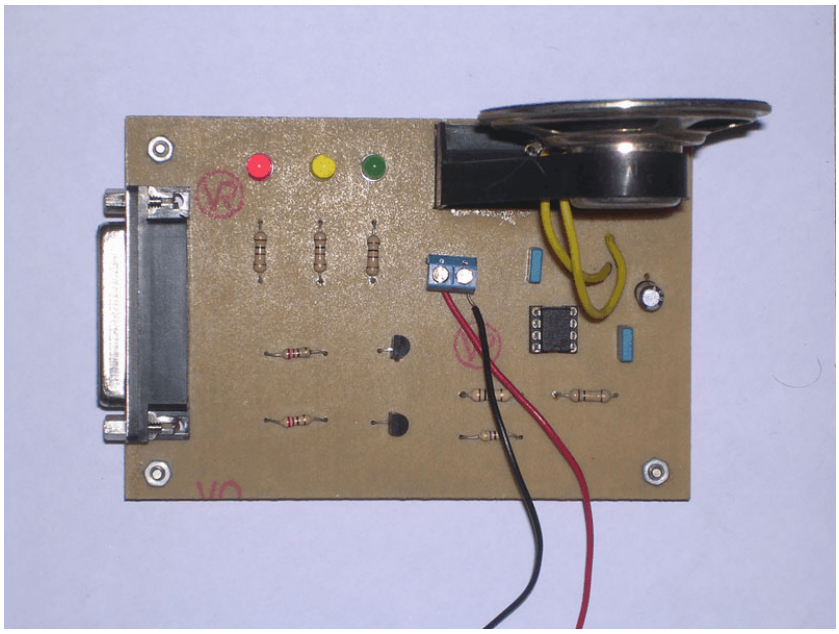


Figura 6.19.”Layout” del circuito para imprimir en un PCB

En la siguiente imagen se muestra como quedo el circuito una vez que se paso a PCB, se taladró y se soldaron los componentes pertinentes.



6.3.2.3 Control de la placa

Para el control tanto del semáforo como del generador de tonos se ha utilizado el puerto paralelo debido a que es un interfaz fácilmente controlable a través del SO Linux y que además no requiere de una configuración compleja en la placa. A continuación hablaremos de como funciona el puerto paralelo bajo Linux y de como enviarle datos.

Reseña histórica del puerto paralelo y la configuración de sus pines

Los puertos paralelos pueden ser usados para conectar una multitud de componentes periféricos:

- Impresoras
- Escaners
- Quemadores de CD
- Discos duros externos
- Iomega Zip removable drives
- Adaptadores de Red
- etc

Figura 6.20. Puerto paralelo de un ordenador

Los puertos paralelos fueron desarrollados originalmente por IBM como una forma de conectar una impresora a la PC. Cuando IBM estaba en el proceso de diseño de los PC´s, la compañía quería que la computadora trabajara con impresoras ofrecidas por Centronics, una empresa líder en la fabricación de impresoras en ese tiempo. IBM decidió no usar el mismo puerto de interfase que Centronics usaba en sus impresoras.

En lugar de eso, los ingenieros de IBM acoplaron un conector de 25 pines, el DB-25, con un conector Centronics de 36 pines para crear un cable especial que conectara la impresora con la computadora. Otros fabricantes de impresoras terminaron adoptando la interfase Centronics, haciendo de este extraño cable híbrido un improbable estándar.

Cuando un PC manda datos a una impresora u a otros dispositivos usando el puerto paralelo, esta manda 8 bits de datos (1 byte) a la vez, de forma distinta al puerto serie el cual manda los 8 bits uno detrás de otro por el mismo cable. El puerto paralelo estándar es capaz de mandar de 50 a 100 kilobytes de datos por segundo.

Echemos una mirada mas cercana a lo que hace cada pin cuando utiliza una impresora:

■ Pin 1 lleva la señal de strobe. Esta mantiene un nivel de voltaje comprendido entre 2.8 y 5v, pero cae debajo de 0.5 volts cuando la computadora manda un byte de datos. Esta caída en el voltaje le dice a la impresora que el dato ha sido enviado.

■ Los pines 2 al 9 son usados para mandar el dato. Para indicar que un bit tiene un valor de 1, una carga de 5v es enviada a través del pin correspondiente. Cuando no hay un voltaje en un pin, indica un valor de 0. Esta es una forma simple pero muy efectiva de transmitir información digital.

■ El pin 10 manda la señal de reconocimiento de la impresora a la computadora. Como el pin 1, esta mantiene una carga y cae debajo de 0.5v para indicarle a la

computadora que el dato fue recibido.

■ Si la impresora está ocupada, mandará un 1 por el pin 11, y cambiará a 0, para indicarle a la computadora que está lista para recibir mas datos.

■ La impresora le avisa a la computadora que se quedó sin papel enviando un 1 por el Pin 12.

■ Mientras la computadora este recibiendo voltaje por el pin 13, sabe que el dispositivo está conectado.

■ La computadora manda un una señal de auto suministro de papel a la impresora a través del pin 14.

■ Si la impresora tiene algún problema, manda un 0 al pin 15 para avisarle a la computadora que existe un error.

■ Cuando un nuevo trabajo de impresión está listo, la computadora manda un 0 para inicializar la impresora.

■ El pin 17 es usado por la computadora para tomar como remotamente desconectada la impresora. Esto se logra enviando 5 volts a la impresora y manteniéndolo así el tiempo que quieras considerarla desconectada.

■ Los Pine 18 al 25 son tierras y son usadas como señales de referencia para el nivel bajo (debajo de 0.5volts).

Figura 6.21. Pines puerto paralelo

DB 25		Centronics 36	
Pin	Signal	Pin	Signal
1	Strobe	1	Strobe
2	data0	2	data0
3	data1	3	data1
4	data2	4	data2
5	data3	5	data3
6	data4	6	data4
7	data5	7	data5
8	data6	8	data6
9	data7	9	data7
10	Acknowledge	10	Acknowledge
11	Busy	11	Busy
12	Paper End	12	Paper End
13	Select	13	Select
14	Auto Feed	14	Auto Feed
15	Error	15	Error
16	Init	16	Init
17	Select In	17	Select In
18	GND	18	GND
19	GND	19	GND
20	GND	20	GND
21	GND	21	GND
22	GND	22	GND
23	GND	23	GND
24	GND	24	GND
25	GND	25	GND
©2000 How Stuff Works		26	GND
		27	GND
		28	GND
		29	GND
		30	GND
		31	Init
		32	Error
		33	Ground
		34	NC
		35	NC
		36	Select In

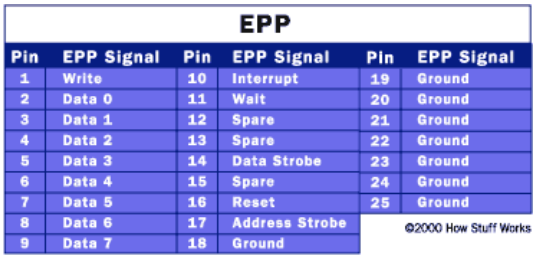
Figura 6.22. Pines DB25 y Centronics 36

Podemos observar como los primeros 25 pines del cable Centronics coinciden con los pines del primer conector.

SPP/EPP/ECP

La especificación original para el puerto paralelo era unidireccional, esto quiere decir que la información solamente puede viajar en una dirección por cada pin. Con la introducción del PS/2 en 1987, IBM ofreció un nuevo diseño de puerto paralelo bidireccional. Este modo es comúnmente conocido como Puerto paralelo estandar (SPP de Standard Parallel Port) y ha reemplazado completamente el diseño original.

La comunicación bidireccional permite a cada dispositivo recibir datos así como también transmitir. Muchos dispositivos usan los 8 pines (del 2 al 9) originalmente diseñados para datos. Usando los mismos 8 pines limita la comunicación a half-duplex, es decir que la información solamente puede viajar en una dirección a la vez. Pero los pines 18 al 25, originalmente utilizados como tierras, pueden ser usados como pins de datos también. Esto permite la comunicación full-duplex (ambas direcciones al mismo tiempo).

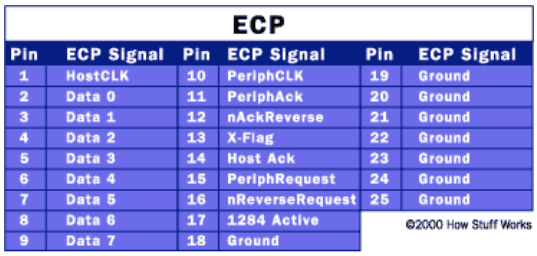


EPP					
Pin	EPP Signal	Pin	EPP Signal	Pin	EPP Signal
1	Write	10	Interrupt	19	Ground
2	Data 0	11	Wait	20	Ground
3	Data 1	12	Spare	21	Ground
4	Data 2	13	Spare	22	Ground
5	Data 3	14	Data Strobe	23	Ground
6	Data 4	15	Spare	24	Ground
7	Data 5	16	Reset	25	Ground
8	Data 6	17	Address Strobe		
9	Data 7	18	Ground		

Figura 6.23 Estándar. EPP

El Puerto Paralelo Mejorado (EPP de Enhanced Parallel Port) fue creado por Intel, Xircom y Zenith en 1991. El EPP permite transmitir mas información cada segundo (500 kilobytes a 2 megabytes). Este fue diseñado para dispositivos que no son impresoras, que se conectarían a este puerto, particularmente dispositivos de almacenamiento, los cuales necesitan la mas alta velocidad de transferencia.

Casi al mismo tiempo de la introducción del EPP, Microsoft y Hewlett Packard conjuntamente anunciaron una especificación llamada salida paralela con capacidad de expansión (ECP de Extended Capabilities Port) en 1992. Mientras el EPP estaba siendo usado para otros dispositivos, el ECP fue diseñado para mejorar la velocidad y funcionalidad de las impresoras.



ECP					
Pin	ECP Signal	Pin	ECP Signal	Pin	ECP Signal
1	HostCLK	10	PeriphCLK	19	Ground
2	Data 0	11	PeriphAck	20	Ground
3	Data 1	12	nAckReverse	21	Ground
4	Data 2	13	X-Flag	22	Ground
5	Data 3	14	Host Ack	23	Ground
6	Data 4	15	PeriphRequest	24	Ground
7	Data 5	16	nReverseRequest	25	Ground
8	Data 6	17	1284 Active		
9	Data 7	18	Ground		

Figura 6.24. Estándar ECP

En 1994, el estándar IEEE 1284 salió en vigencia. Este incluía las dos especificaciones para los dispositivos para puerto paralelo, EPP y ECP. Para que estos funcionen, tanto el sistema operativo como el dispositivo deben soportar la especificación requerida. Esto ya no es un problema en estos días, ya que la mayoría de las computadoras soportan SPP, ECP y EPP y detectan que modo necesita ser utilizado, dependiendo del dispositivo conectado. Si queremos variar el modo de funcionamiento podemos hacerlo a través de la BIOS de la mayoría de los ordenadores..

Manejo del puerto paralelo en Línux

La dirección base llamada BASE, puede ser: 0x378, 0x278, 0x3bc. Dependiendo del puerto a utilizar (/dev/lp0, /dev/lp1, /dev/lp2).

El puerto BASE llamado también como el Data Port controla las señales de datos D0..D7, cada señal es un bit [0,1] que corresponde a los valores de tensión 0v y +5v. Cuando escribimos en el puerto el valor se queda "congelado" en los pines D0..D7

El puerto BASE+1 llamado el Status Port el cual es de sólo lectura, nos permite leer el estado del puerto, este puerto se compone como el anterior de 8 señales:

- 0 RESERVADO
- 1 RESERVADO
- 2 IRQ STATUS
- 3 ERROR
- 4 SLCT
- 5 PE
- 6 ACK
- 7 BUSY

A parte de estos dos hay un tercero llamado Control Port que es de sólo escritura, si se intenta leer nos devuelve lo último que hemos enviado por este puerto. Este puerto tiene la dirección BASE+2 y consta también de 8 bits.

Bit	**Nombre**	**Descripción**	**Tipo**
0	STROBE	Indica a la impresora que la información está completa y que puede imprimir el carácter.	(Lógica negativa)
1	AUTO_FDXT	Señal de autoalimentación. Controla la forma de manejar los saltos de línea.	(Lógica negativa)
2	INIT	Reinicializa la impresora.	(Lógica positiva)
3	SCLT_IN	Señal para indicar a la impresora que se ponga en On-Line.	(Lógica negativa)
4	Enable the PPA IRQ	Ocurre cuando e produce una transacción ACK a bajo nivel.	(Lógica positiva)

5	Extended mode direction	0= Rrite,1= Read	(Lógica positiva)
6 y 7	RESERVADOS		

Para utilizar las funciones de manejo de I/O mapeadas se tiene que tener en cuenta que se tiene que dar permiso a un rango de direcciones a las cuales se tiene que acceder, para ello utilizamos la funcion ioperm. No nos extrañemos si nos falla el programa y nos da un error de Permission denied eso sera por no dar permiso al I/O deseado.

```
/*El parametro rango es para indicar el rango de direcciones a dar permiso y el
parametro activar 1 = Dar permiso o 0 = Quitar permiso.*/

    if(ioperm(BaseAddr,3,1)) {
      perror("Dando permisos");
      exit(1);
    }
        .
        .
    // Utilizamos el BaseAddr
        .
        .

    if(ioperm(BaseAddr,3,0)) {
      perror("Quitando permisos");
      exit(1);
    }
```

Una vez dado el permiso al rango de direcciones a utilizar, se podrán utilizar funciones del estilo outb inb, etc ... Para poder usar estas funciones de bajo nivel se tiene que incluir el header sys/io.h y para compilar el programa utilizaremos el parámetro -O2.

```
/*Enviar el carácter FORM-FEED (Salto página) a la impresora.*/

#include <stdio.h>
#include <sys/io.h>
#define direccio_pp 0x378

int main(void)
{

   ioperm(direccio_pp,1,1);  // Damos permiso a 1 direccion

   outb(12,direccio_pp);     // Enviamos el caracter 12 (Page-Feed)

   ioperm(direccio_pp,1,0);  // Quitamos el permiso
}

/*Otra manera de hacerlo a mas alto nivel seria utilizando las funciones de open,write, etc...*/
```

6.4 Implementación del proceso Cliente

El cliente ha sido programado para funcionar en el sistema operativo Linux sobre un dispositivo con procesador XSCALE con tecnología inalámbrica Wi-fi. El lenguaje de programación empleado ha sido C puro.

El cliente no se ha podido programar en Java debido a que la máquina virtual propietaria de Sun para este tipo de dispositivos móviles no son gratuitas como en el caso de los PC`s habituales. Además C permite una programación a más bajo nivel y sobre todo no requiere de una máquina virtual que consuma un exceso de recursos como en el caso de Java.

En el cliente se distinguen dos procesos: el proceso padre y un proceso hijo. El proceso padre es el encargado de generar un proceso hijo que realizará las operaciones que se describen en el apartado de funcionamiento del cliente, y de esperar a que este hijo muera para generar otro nuevo. La figura 6.25 ilustra como se lleva a cabo la generación de hijos por parte del proceso padre.

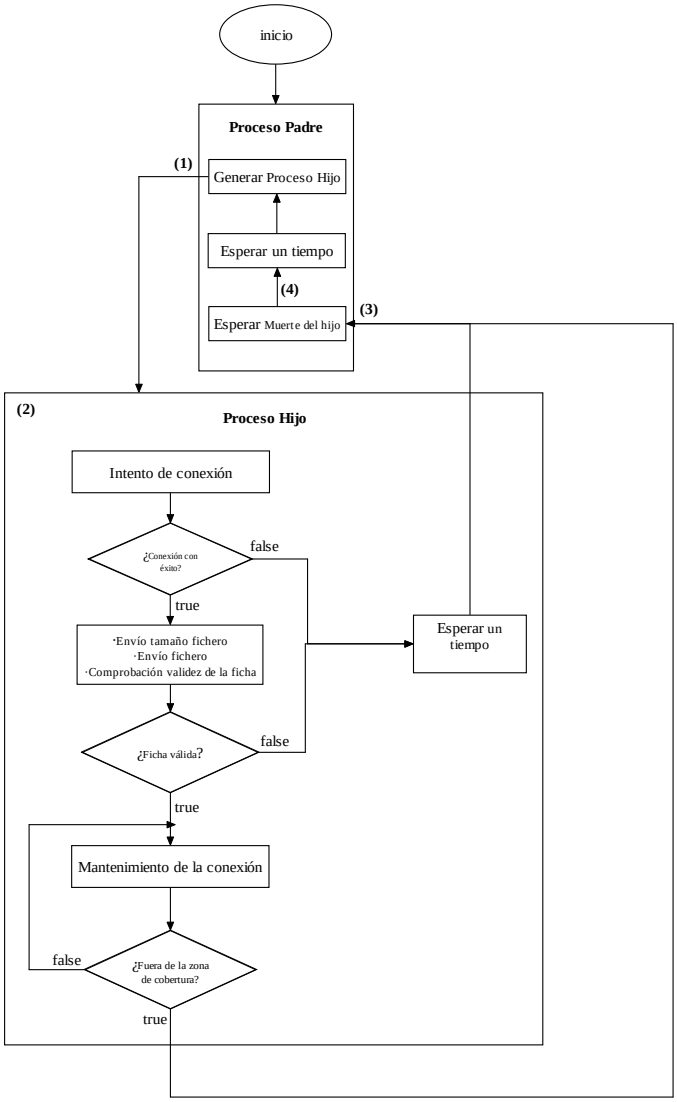


Figura 6.25. Implementación proceso cliente

En un primer lugar (1) el proceso padre genera un proceso hijo para que este realice el funcionamiento (2) que se describió en el apartado de funcionamiento del proceso cliente. En el caso de que no pueda conectarse con ningún servidor, de que el cliente salga de la zona de cobertura o la ficha no sea válida el proceso padre esperará la "muerte" del proceso hijo (3) para posteriormente volver a generar un nuevo proceso hijo (4).

El cliente funciona sobre TCP/IP, por tanto la información que se transmite en todo momento se hace sobre el nivel de aplicación como se especifica en la figura 6.26.

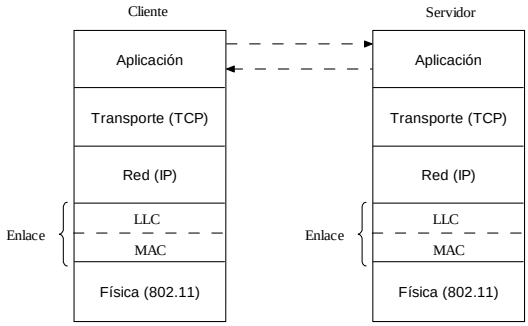


Figura 6.26. Capa en la que funciona el programa

6.5 Implementación del módulo servidor toma de decisión

El proceso toma de decisión se ha programado también en C puro para funcionar con un procesador XSCALE sobre un sistema operativo Linux. Es un proceso servidor implementado a través de sockets locales que atiende las peticiones del procesador XML.

Se han utilizado sockets locales por motivos de seguridad, de esta forma al no existir direcciones IP se hace mas difícil que un intruso pudiera tomar el control de este proceso.

Capítulo 7

Manual de usuario

7.1 Introducción

En este manual se explica de forma detallada la instalación y configuración para hacer funcionar el sistema. También se mostrarán capturas de pantalla del sistema funcionando. Se mostrará como instalar el compilador cruzado gcc 3.0 y como configurar el sistema operativo Linux para que los programas funcionen sobre una arquitectura XSCALE. Se describirá también los archivos que contiene cada programa, que contiene cada uno así como la forma de transferirlos a los dispositivos.

7.2 Instalación y modo de uso del compilador cruzado gcc 3.0

El proceso a aseguir es muy sencillo, únicamente hay que descomprimir el fichero al modo habitual. Generalmente, en los archivos README e INSTALL podremos encontrar información especifica para el caso en el que estemos.

Para el caso que nos ocupa, estableceremos en primer lugar una serie de variables de entorno para definir que programas se usaran como compilador, linkador, ensamblador, etc...

```
export
AR="/usr/local/arm/2.95.3/bin/arm-linux-ar"export
CC="/usr/local/arm/2.95.3/bin/arm-linux-gcc"
export AS="/usr/local/arm/2.95.3/bin/arm-linux-as"
export LD="/usr/local/arm/2.95.3/bin/arm-linux-ld"
```

Las rutas indicadas son sólo a modo de ejemplo y deberán reflejar la situación real de los ejecutables a usar.

Problemas con las compilación cruzada

Anteriormente se ha explicado el procedimiento a seguir para su compilación e instalación, pero se ha supuesto que no se iba a presentar ningún problema. A continuación, se enumeran una serie de problemas comunes y la manera de solucionarlos.

■ No se encuentran las librerías o las cabeceras

Este problema suele venir a consecuencia de un uso inapropiado del argumento --with-bluez. Este argumento debe indicar un directorio con dos subdirectorios, lib e include, en los que se encontraran librerías y cabeceras respectivamente. Tras ejecutar el comando make, las librerías están en un subdirectorio oculto dentro del subdirectorio src, mientras que las cabeceras están en el directorio include. Se recomienda encarecidamente indicar un lugar de instalación con --prefix.

■ No se ha definido PATH_MAX

Parece ser que existe un error, y es necesario incluir la cabecera limits.h en un par de archivos. Para corregirlo incluiremos la siguiente linea en los archivos tools/hciconfig.c y hidd/sdp.c:

```
#include <linux/limits.h>
```

7.3 Manual del cliente sobre plataforma XSCALE

El programa cliente está implementado en el archivo cliente.c. La compilación es la habitual del compilador cruzado.

El cliente se transfiere a la PDA a través de múltiples modos como por ejemplo: el cliente ftp OpieFtp, a través de scp o también mediante wget, teniendo que tener instalado un servidor web como Apache.

En la siguiente captura se muestra una imagen del programa cliente corriendo.

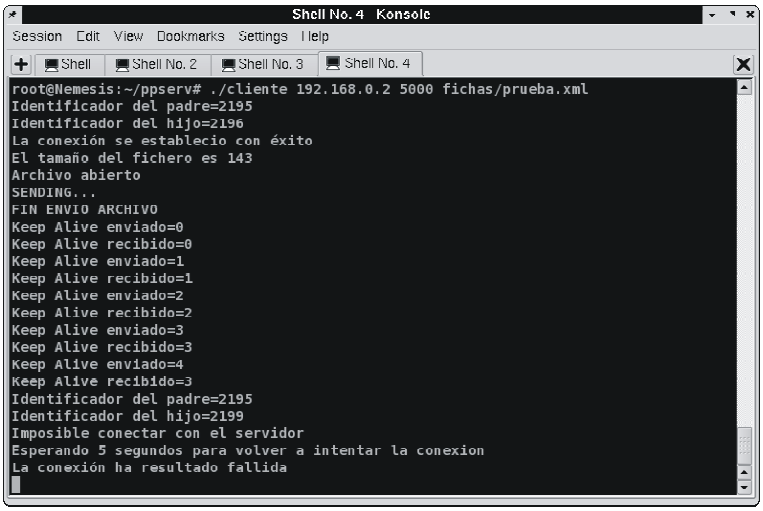


Figura 7.1. Captura del cliente en ejecución

7.4 Manual de toma de decisión sobre plataforma XSCALE

El programa toma de decisión posee cuatro archivos:

· ppserv.c : que contiene la implementación del programa en su totalidad haciendo llamadas a los dos archivos que se describen a continuación.

· tomadecision.c : es el lugar donde obtiene la deficiencia o deficiencias que tiene un usuario que entra al sistema a partir del número que es potencia de dos o suma de número potencias de dos que nos pasa el procesador XML. Una vez obtenida la deficiencia se indican las acciones a realizar en función de cual sea ésta.

· constantes.c : es el lugar donde se definen las deficiencias con las que funciona el sistema, así como la temporización en rojo que asignamos a cada una de ellas. También se define la temporización del semáforo en estado normal.

· Makefile

Según lo anterior si queremos añadir una nueva deficiencia, debemos de añadirla al fichero constantes.c asignandole un número potencia de dos que no éste ya en uso y dando una temporización en el estado rojo del semáforo. Posteriormente debemos de modificar de forma adecuada el fichero tomadecision.c.

La compilación se realiza a través del comando Make.

En la siguiente captura se muestra una imagen del programa ppserv corriendo.

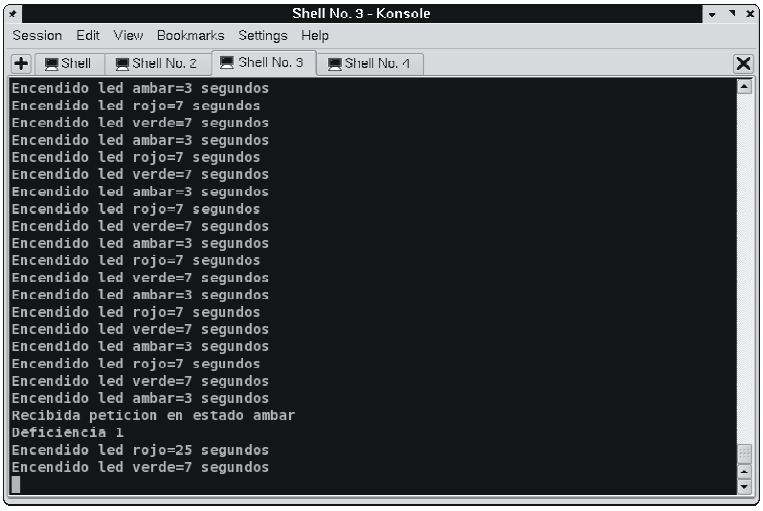


Figura 7.2. Captura del programa ppserv en ejecución

Capítulo 8

Pruebas

8.1 Equipos utilizados para las pruebas

- Dongle Bluetooth
- PDA iPAQ 5550
- Sony Ericsson P800
- Tarjeta Wifi
- Mini-ordenador SB-X255 de Compulab con S.O Linux
- PC habitual con S.O Linux

8.2 Configuración del sistema

Se conectaron y situaron los equipos tal y como se muestra en la figura. Se puede observar que la antena wireless se coloco en posición de máxima directividad en relación a la posición de los dispositivos inalámbricos. Por otro lado, se utilizo un pasillo del laboratorio a modo de paso cebra.

Figura 8.1. Equipo de pruebas A

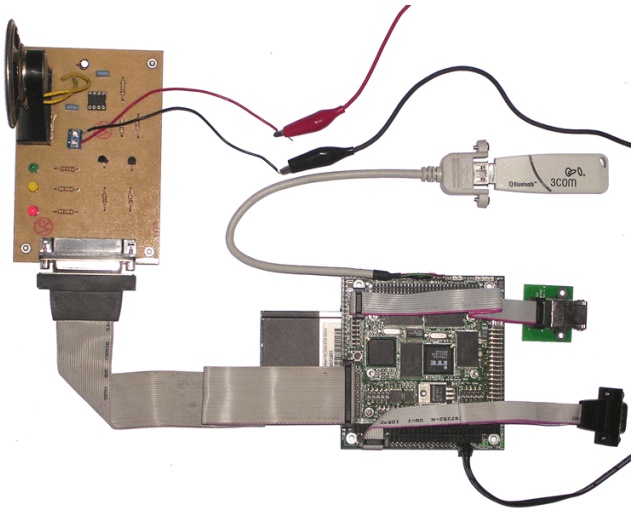


Figura 8.2. Equipo de pruebas B

8.3 Pruebas

Para las pruebas con respecto a los tiempos se comprobó que una deficiencia locomotriz de grado uno podría tener un factor multiplicativo de cuatro en cuanto al tiempo en relación a la temporización normal del sistema. Por otro lado, una deficiencia de grado dos podría tener un factor multiplicativo de aproximadamente tres, mientras que una de grado tres un factor multiplicativo de dos.

Se comprobó que como era de esperar el alcance de los Bluetooth era de 10 metros aproximadamente (a esta distancia salían de la zona de cobertura) mientras que el wifi alcanzaba unos 30 metros. La disminución de las distancias se debe a las reflexiones y difracciones que se producían en el entorno de pruebas, al no utilizar como medio de transmisión el espacio libre.

Figura 8.3. PDA usada en las pruebas

El sistema también fue sometido a una prueba con varios usuarios, donde se comprobó que éste respondía correctamente incluso con una gran demanda de peticiones, haciendo prevalecer de esta forma aquella el caso más restrictivo.

Figura 8.4. Móvil usado en las pruebas

El sistema no presentó ningún problema por estar en funcionamiento con dos tecnologías distintas como son Bluetooth y Wifi, que son dos tecnologías que funcionan a la misma frecuencia de 2.4 Ghz.

Por otro lado el sistema en Bluetooth no estaba dotado de encriptación, quedando este último para proyectos futuros. En wifi se encriptó por WEP, pero sería mejor una técnica de encriptado como SSL, ya que de esta forma nos ahorraríamos muchos problemas si el sistema se llegase a desarrollar en un entorno real.

Un problema que surgió y que queda para un desarrollo posterior de este proyecto fue el largo alcance de los dispositivos. Como consecuencia de ésto, el sistema se activa aunque el discapacitado esté a varios a metros del semáforo. Una solución sería el diseño de una antena que se adecuada para nuestro proposito tal y cómo se explica en líneas futuras.

Conclusión y líneas futuras

Este proyecto ha consistido en la elaboración de un sistema capaz de regular el paso de una vía regida por un semáforo. El sistema ha sido desarrollado con la intención de ser utilizados para el beneficio de personas que sufren algún tipo de discapacidad, y a las cuales el mero hecho de cruzar una vía o tantear donde está el botón que hace cambiar el estado del semáforo a verde les supone un gran esfuerzo.

Ya que el sistema está destinado a este tipo de personas unos de los principales esfuerzos lo hemos centrado en conseguir que el sistema sea los más automático posible, para liberar así al usuario de la tediosa tarea de interactuar con el dispositivo móvil.

Este sistema ha consistido en el desarrollo de una aplicación cliente-servidor. La parte del cliente se desarrolló para dos tipos de dispositivos inalámbricos distintos (móvil y PDA) usando también dos tecnologías inalámbricas distintas para cada uno (Bluetooth y Wi-FI). La parte del servidor se desarrollo pará un mini ordenador el cual por su pequeño tamaño seria fácilmente utilizable para su funcionamiento en semáforos. Este también está adaptado para servir a clientes por medio de las dos tecnologías anteriormente comentadas.

Lineas futuras

El presente proyecto deja puertas abiertas a futuras investigaciones, debido a la continua y rápida evolución que experimentan todos los tipos de redes inalámbricas. Algunas posibles investigaciones son:

- Añadir encriptación a las comunicaciones inalámbricas
- Posibilidad de modificar la potencia de transmisión de las tarjetas wifi
- Diseño de una antena adecuada para el funcionamiento del sistema en un entorno real.